

Dirk Deimeke
Stefan Kania
Daniel van Soest
Peer Heinlein
Axel Miesen



CentOS
Debian GNU/Linux
openSUSE Leap
Ubuntu Server LTS



```
$IPT -A int_to_dmz -m st  
$IPT -A int_to_dmz -j ex  
### Verkehr aus der DMZ  
$IPT -A dmz_to_ext -m st  
$IPT -A dmz_to_ext -m st  
-s $MAIL -j ACCEPT  
o_ext -m st  
ACCEPT
```

Linux-Server

Das umfassende Handbuch

- ▶ Linux-Server einrichten und administrieren
- ▶ Backup, Sicherheit, Netzwerke und modernes DevOps
- ▶ Container mit Docker, Automatisierung mit Ansible, Datei-Server und Userverwaltung mit LDAP und Samba

7., aktualisierte Auflage



Mit allen Konfigurationsdateien zum Download



Rheinwerk
Computing

Liebe Leserin, lieber Leser,

selbst der erfahrenste Administrator kommt manchmal in eine Situation, wo er nicht weiter weiß: Ein neuer Dienst muss implementiert werden, aber wie lässt er sich in die bestehende Infrastruktur integrieren? Solche Fragen beantwortet das Handbuch »Linux-Server« schnell und zuverlässig. Das Konzept des Buches ist seit der ersten Auflage unverändert: Es wird für jede Aufgabe, die bei der Administration von Linux-Servern anfällt, mindestens ein Tool vorgestellt, mit dem Sie die betreffende Aufgabe sicher lösen können. Berücksichtigt werden dabei die gängigen Linux-Distributionen.

Die Autoren sind erfahrene Linux-Experten mit langjähriger Erfahrung in mittleren bis sehr großen IT-Infrastrukturen. Jede Auflage wird von ihnen gründlich aktualisiert: Neue Themen werden aufgenommen und die zahlreichen Leserrückmeldungen fließen in die Überarbeitung ein. Und so bietet Ihnen auch diese siebte Auflage wieder topaktuelles Wissen zur Administration von Linux-Servern. Dirk Deimeke, Daniel van Soest, Stefan Kania, Peer Heinlein und Axel Miesen geben Ihnen neben dem benötigten Hintergrundwissen auch viele Tipps für die Praxis mit auf den Weg. Außerdem stellen die Autoren Ihnen zahlreiche geprüfte Beispielskripte und Konfigurationen zur Verfügung, die Sie für Ihre Aufgaben nutzen können.

Zuletzt noch ein Hinweis in eigener Sache: Das Buch wurde mit großer Sorgfalt geschrieben, lektoriert und produziert. Sollte dennoch etwas nicht so funktionieren, wie Sie es erwarten, dann setzen Sie sich bitte direkt mit mir in Verbindung. Ihre Anregungen und Fragen sind jederzeit willkommen.

Ihr Christoph Meister

Lektorat Rheinwerk Computing

christoph.meister@rheinwerk-verlag.de

www.rheinwerk-verlag.de

Rheinwerk Verlag · Rheinwerkallee 4 · 53227 Bonn

Hinweise zur Benutzung

Dieses E-Book ist **urheberrechtlich geschützt**. Mit dem Erwerb des E-Books haben Sie sich verpflichtet, die Urheberrechte anzuerkennen und einzuhalten. Sie sind berechtigt, dieses E-Book für persönliche Zwecke zu nutzen. Sie dürfen es auch ausdrucken und kopieren, aber auch dies nur für den persönlichen Gebrauch. Die Weitergabe einer elektronischen oder gedruckten Kopie an Dritte ist dagegen nicht erlaubt, weder ganz noch in Teilen. Und auch nicht eine Veröffentlichung im Internet oder in einem Firmennetzwerk.

Die ausführlichen und rechtlich verbindlichen Nutzungsbedingungen lesen Sie im Abschnitt *Rechtliche Hinweise*.

Dieses E-Book-Exemplar ist mit einem **digitalen Wasserzeichen** versehen, einem Vermerk, der kenntlich macht, welche Person dieses Exemplar nutzen darf:

Exemplar Nr. xbgf-85zp-rayj-vidm
zum persönlichen Gebrauch für
Thomas Bartholomäus,
minduxde,
thomas.bartholomaeus@mindux.de

Impressum

Dieses E-Book ist ein Verlagsprodukt, an dem viele mitgewirkt haben, insbesondere:

Lektorat Christoph Meister

Korrektorat Friederike Daenecke, Zülpich

Herstellung E-Book Stefanie Meyer, Norbert Englert

Covergestaltung Bastian Illerhaus

Coverbild iStock: 918951042 © monsitj, 522512159 © Squaredpixels; Adobe: 252170100
© xiaoliangge; Pinguin Tux: lewing@isc.tamu.edu Larry Ewing and The GIMP

Satz E-Book Daniel van Soest

Wir hoffen sehr, dass Ihnen dieses Buch gefallen hat. Bitte teilen Sie uns doch Ihre Meinung mit und lesen Sie weiter auf den [Serviceseiten](#).

Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.dnb.de> abrufbar.

ISBN 978-3-8362-9616-8 (E-Book)

ISBN 978-3-8362-9618-2 (Bundle)

7., aktualisierte Auflage 2023

© Rheinwerk Verlag GmbH, Bonn 2023

www.rheinwerk-verlag.de

Inhalt

Vorwort	33
Über dieses Buch	43

1 Der Administrator 47

1.1 Der Beruf des Systemadministrators	47
1.1.1 Berufsbezeichnung und Aufgaben	47
1.1.2 Job-Definitionen	48
1.1.3 Definitionen der Management-Level	52
1.2 Nützliche Fähigkeiten und Fertigkeiten	54
1.2.1 Soziale Fähigkeiten	54
1.2.2 Arbeitstechniken	55
1.3 Das Verhältnis des Administrators zu Normalsterblichen	57
1.3.1 Der Chef und andere Vorgesetzte	57
1.3.2 Benutzer	58
1.3.3 Andere Administratoren	58
1.4 Unterbrechungsgesteuertes Arbeiten	59
1.5 Einordnung der Systemadministration	60
1.5.1 Arbeitsgebiete	60
1.5.2 DevOps	62
1.6 Ethischer Verhaltenskodex	64
1.7 Administration – eine Lebenseinstellung?	65

TEIL I Grundlagen

2 Der Bootvorgang 69

2.1 Der Bootloader GRUB 2	69
2.1.1 Funktionsweise	69
2.1.2 Installation	70
2.1.3 Konfiguration	70
2.2 Bootloader Recovery	76

2.3	Der Kernel und die initrd	77
2.3.1	initrd erstellen und modifizieren	78
2.3.2	initrd manuell modifizieren	82
2.4	systemd	83
2.4.1	Begriffe	84
2.4.2	Kontrollieren von Diensten	85
2.4.3	Aktivieren und Deaktivieren von Diensten	87
2.4.4	Erstellen und Aktivieren eigener Service Units	88
2.4.5	Target Units	90
2.4.6	»systemd«- und Servicekonfigurationen	91
2.4.7	Anzeige von Dienstabhängigkeiten	92
2.4.8	Logs mit journald	94
2.4.9	Abschlussbemerkung	95
3	Festplatten und andere Devices	97
3.1	RAID	97
3.1.1	RAID-0	98
3.1.2	RAID-1	98
3.1.3	RAID-5	98
3.1.4	RAID-6	99
3.1.5	RAID-10	99
3.1.6	Zusammenfassung	100
3.1.7	Weich, aber gut: Software-RAID	101
3.1.8	Software-RAID unter Linux	102
3.1.9	Abschlussbemerkung zu RAIDs	109
3.2	Rein logisch: Logical Volume Manager (LVM)	110
3.2.1	Grundlagen und Begriffe	112
3.2.2	Setup	113
3.2.3	Aufbau einer Volume Group mit einem Volume	114
3.2.4	Erweiterung eines Volumes	117
3.2.5	Eine Volume Group erweitern	118
3.2.6	Spiegelung zu einem Volume hinzufügen	119
3.2.7	Eine defekte Festplatte ersetzen	120
3.2.8	Backups mit Snapshots	121
3.2.9	Mirroring ausführlich	125

3.2.10	Thin Provisioning	129
3.2.11	Kommandos	132
3.3	udev	133
3.3.1	udev-Regeln	133
3.3.2	Eigene Regeln schreiben	134
3.4	Alles virtuell? »/proc«	137
3.4.1	CPU	137
3.4.2	RAM	138
3.4.3	Kernelkonfiguration	139
3.4.4	Kernelparameter	140
3.4.5	Gemountete Dateisysteme	140
3.4.6	Prozessinformationen	141
3.4.7	Netzwerk	142
3.4.8	Änderungen dauerhaft speichern	143
3.4.9	Abschlussbemerkung	143

4 Dateisysteme 145

4.1	Dateisysteme: von Bäumen, Journalen und einer Kuh	145
4.1.1	Bäume	146
4.1.2	Journalen	148
4.1.3	Und die Kühe? COW-fähige Dateisysteme	148
4.2	Praxis	149
4.2.1	Ext2/3-FS aufgebohrt: mke2fs, tune2fs, dumpe2fs, e2label	149
4.2.2	ReiserFS und seine Tools	152
4.2.3	XFS	153
4.2.4	Das Dateisystem vergrößern oder verkleinern	154
4.2.5	BtrFS	155
4.3	Fazit	162

5 Berechtigungen 163

5.1	User, Gruppen und Dateisystemstrukturen	163
5.2	Dateisystemberechtigungen	166
5.2.1	Spezialbits	167

5.3	Erweiterte POSIX-ACLs	170
5.3.1	Setzen und Anzeigen von einfachen ACLs	171
5.3.2	Setzen von Default-ACLs	173
5.3.3	Setzen von erweiterten ACLs	175
5.3.4	Entfernen von ACLs	177
5.3.5	Sichern und Zurückspielen von ACLs	178
5.4	Erweiterte Dateisystemattribute	179
5.4.1	Attribute, die jeder Benutzer ändern kann	179
5.4.2	Attribute, die nur »root« ändern kann	180
5.4.3	Weitere Attribute	181
5.5	Quotas	181
5.5.1	Installation und Aktivierung der Quotas	182
5.5.2	Journaling-Quotas	183
5.5.3	Quota-Einträge verwalten	184
5.6	Pluggable Authentication Modules (PAM)	188
5.6.1	Verschiedene PAM-Typen	189
5.6.2	Die PAM-Kontrollflags	189
5.6.3	Argumente zu den Modulen	190
5.6.4	Modulpfade	190
5.6.5	Module und ihre Aufgaben	191
5.6.6	Die neuere Syntax bei der PAM-Konfiguration	192
5.7	Konfiguration von PAM	194
5.8	ulimit	195
5.8.1	Setzen der ulimit-Werte	196
5.9	Abschlussbemerkung	197

TEIL II Aufgaben

6	Paketmanagement	201
6.1	Paketverwaltung	201
6.1.1	rpm oder deb?	202
6.1.2	dnf, yast, zypper oder apt?	204
6.1.3	Außerirdische an Bord – alien	205

6.2	Pakete im Eigenbau	206
6.2.1	Vorbereitungen	207
6.2.2	Am Anfang war das Makefile	207
6.2.3	Vom Fellknäuel zum Paket	210
6.2.4	Patchen mit patch und diff	214
6.2.5	Updates sicher konfigurieren	217
6.3	Updates nur einmal laden: Cache	219
6.3.1	deb-basierte Distributionen: apt-cacher-ng	219
6.3.2	Installation	219
6.3.3	Konfiguration	220
6.3.4	Clientkonfiguration	222
6.3.5	Fütterungszeit – bereits geladene Pakete dem Cache hinzufügen	222
6.3.6	Details: Report-HTML	223
6.3.7	rpm-basierte Distributionen	223
6.4	Alles meins: Mirror	224
6.4.1	deb-basierte Distributionen: debmirror	224
6.4.2	Konfiguration	224
6.4.3	Benutzer und Gruppe anlegen	224
6.4.4	Verzeichnisstruktur anlegen	225
6.4.5	Mirror-Skript erstellen (Ubuntu)	225
6.4.6	Cronjobs einrichten	228
6.4.7	Schlüssel importieren	228
6.4.8	Mirror erstellen	229
6.4.9	Mirror verfügbar machen – Webdienst konfigurieren	229
6.4.10	Clientkonfiguration	230
6.4.11	rpm-basierte Distributionen	230
6.4.12	Benutzer und Gruppe anlegen	231
6.4.13	Verzeichnisstruktur anlegen: openSUSE Leap	231
6.4.14	Verzeichnisstruktur anlegen: CentOS	231
6.4.15	Mirror-Skript erstellen	232
6.4.16	Cronjobs einrichten	233
6.4.17	Mirror erstellen	234
6.4.18	Mirror verfügbar machen – Webdienst konfigurieren	234
6.4.19	Clientkonfiguration: openSUSE Leap	235
6.4.20	Clientkonfiguration: CentOS	236

7 Backup und Recovery 237

7.1 Backup gleich Disaster Recovery?	237
7.2 Backupstrategien	238
7.3 Datensicherung mit tar	241
7.3.1 Weitere interessante Optionen für GNU-tar	242
7.3.2 Sicherung über das Netzwerk mit tar und ssh	243
7.4 Datensynchronisation mit rsync	243
7.4.1 Lokale Datensicherung mit rsync	244
7.4.2 Synchronisieren im Netzwerk mit rsync	244
7.4.3 Wichtige Optionen für rsync	245
7.4.4 Backupsript für die Sicherung auf einen Wechseldatenträger	246
7.4.5 Backupsript für die Sicherung auf einen Backupserver	247
7.4.6 Verwendung von ssh für die Absicherung von rsync	249
7.5 Imagesicherung mit dd	250
7.5.1 Sichern des Master Boot Records (MBR)	251
7.5.2 Die Partitionstabelle mithilfe von dd zurückspielen	251
7.5.3 Images mit dd erstellen	252
7.5.4 Einzelne Dateien mit dd aus einem Image zurückspielen	252
7.5.5 Abschlussbemerkung zu dd	254
7.6 Disaster Recovery mit ReaR	255
7.6.1 ReaR installieren	256
7.6.2 ReaR konfigurieren	256
7.6.3 Aufrufparameter von ReaR	258
7.6.4 Der erste Testlauf	259
7.6.5 Der Recovery-Prozess	263
7.6.6 Die ReaR-Konfiguration im Detail	265
7.6.7 Migrationen mit ReaR	266

TEIL III Dienste

8 Webserver 271

8.1 Apache	271
8.1.1 Installation	271
8.1.2 Virtuelle Hosts einrichten	272
8.1.3 Debian/Ubuntu: Virtuelle Hosts aktivieren	274

8.1.4	HTTPS konfigurieren	275
8.1.5	Apache-Server mit ModSecurity schützen	280
8.1.6	Tuning und Monitoring	285
8.2	nginx	289
8.2.1	Installation	289
8.2.2	Grundlegende Konfiguration	290
8.2.3	Virtuelle Hosts	290
8.2.4	HTTPS mit nginx	293
8.3	PHP	294
8.3.1	Installation	294
8.3.2	PHP in den Webseitenkonfigurationen aktivieren	297
8.3.3	Funktionstest	299
8.3.4	Tipps und Tricks	299
8.4	Fortgeschrittene TLS-Konfiguration und Sicherheitsfunktionen	301
8.4.1	SSL/TLS	301
8.4.2	Konfiguration in Apache2	302
8.4.3	Konfiguration in nginx	304
8.4.4	Informationen und Anregungen	305
9	FTP-Server	307
9.1	Einstieg	307
9.1.1	Das File Transfer Protocol	307
9.1.2	vsftpd	308
9.2	Download-Server	308
9.3	Zugriff von Usern auf ihre Homeverzeichnisse	310
9.4	FTP über SSL (FTPS)	311
9.5	Anbindung an LDAP	313
10	Mailserver	315
10.1	Postfix	315
10.1.1	Installation der Postfix-Pakete	316
10.1.2	Grundlegende Konfiguration	316
10.1.3	Postfix als Relay vor Exchange, Dovecot oder anderen Backends	319

10.1.4	Die Postfix-Restrictions: Der Schlüssel zu Postfix	321
10.1.5	Weiterleitungen und Aliasse für Mailadressen	330
10.1.6	SASL/SMTP-Auth	331
10.1.7	SSL/TLS für Postfix einrichten	333
10.2	POP3/IMAP-Server mit Dovecot	335
10.2.1	Installation der Dovecot-Pakete	335
10.2.2	Vorbereitungen im Linux-System	336
10.2.3	Log-Meldungen und Debugging	336
10.2.4	User-Authentifizierung	337
10.2.5	Aktivierung des LMTP-Servers von Dovecot	339
10.2.6	Einrichten von SSL/TLS-Verschlüsselung	340
10.2.7	Der Ernstfall: Der IMAP-Server erwacht zum Leben	341
10.2.8	Dovecot im Replikations-Cluster	342
10.2.9	Einrichtung der Replikation	343
10.2.10	Hochverfügbare Service-IP	346
10.3	Anti-Spam/Anti-Virus mit Rspamd	348
10.3.1	Mails ablehnen oder in die Quarantäne filtern?	348
10.3.2	Installation von Rspamd, ClamAV und Redis	349
10.3.3	Update der Virensignaturen und Start der Dienste	350
10.3.4	Die Architektur von Rspamd	351
10.3.5	Einbindung von Rspamd an Ihren Postfix-Mailserver	352
10.3.6	Konfiguration des Rspamd	354
10.3.7	Konfiguration von Upstream-Quellen	356
10.3.8	Redis als schnelle Datenbank an der Seite von Rspamd	357
10.3.9	Die Definition auszulösender Aktionen	357
10.3.10	Statistik und Auswertung im Webinterface	359
10.3.11	ClamAV in Rspamd einbinden	360
10.3.12	Späteres Filtern über Mail-Header	361
10.3.13	RBLs in Rspamd	362
10.3.14	Bayes in Rspamd	364
10.3.15	Eigene White- und Blacklists führen	365
10.3.16	Einrichtung von DKIM zur Mailsignierung	367
10.3.17	Ausblick: Einbindung weiterer Prüfungsmethoden	370
10.4	Monitoring und Logfile-Auswertung	370

11	Datenbank	371
11.1	MariaDB in der Praxis	371
11.1.1	Installation und grundlegende Einrichtung	371
11.1.2	Replikation	373
11.1.3	Master-Master-Replikation	380
11.2	Tuning	384
11.2.1	Tuning des Speichers	384
11.2.2	Tuning von Indizes	390
11.3	Backup und Point-In-Time-Recovery	394
11.3.1	Restore zum letztmöglichen Zeitpunkt	395
11.3.2	Restore zu einem bestimmten Zeitpunkt	395
12	Syslog	397
12.1	Der Aufbau von Syslog-Nachrichten	397
12.2	systemd mit journalctl	399
12.2.1	Erste Schritte mit dem journalctl-Kommando	400
12.2.2	Filtern nach Zeit	402
12.2.3	Filtern nach Diensten	403
12.2.4	Kernelmeldungen	404
12.2.5	Einrichten eines Log-Hosts	405
12.3	Der Klassiker: Syslogd	408
12.4	Syslog-ng	410
12.4.1	Der »options«-Abschnitt	410
12.4.2	Das »source«-Objekt	412
12.4.3	Das »destination«-Objekt	412
12.4.4	Das »filter«-Objekt	414
12.4.5	Das »log«-Objekt	416
12.5	Rsyslog	416
12.5.1	Eigenschaftsbasierte Filter	416
12.5.2	Ausdrucksbasierte Filter	417
12.6	Loggen über das Netz	418
12.6.1	SyslogD	418
12.6.2	Syslog-ng	419
12.6.3	Rsyslog	420

12.7 Syslog in eine Datenbank schreiben	420
12.7.1 Anlegen der Log-Datenbank	420
12.7.2 In die Datenbank loggen	421
12.8 Fazit	423
13 Proxy-Server	425
<hr/>	
13.1 Einführung des Stellvertreters	425
13.2 Proxys in Zeiten des Breitbandinternets	426
13.3 Herangehensweisen und Vorüberlegungen	427
13.4 Grundkonfiguration	427
13.4.1 Aufbau des Testumfelds	428
13.4.2 Netzwerk	428
13.4.3 Cache	429
13.4.4 Logging	430
13.4.5 Handhabung des Dienstes	432
13.4.6 Objekte	433
13.4.7 Objekttypen	435
13.4.8 Objektlisten in Dateien	435
13.4.9 Regeln	436
13.4.10 Überlagerung mit »first match«	438
13.4.11 Anwendung von Objekten und Regeln	439
13.5 Authentifizierung	440
13.5.1 Benutzerbasiert	443
13.5.2 Gruppenbasiert	452
13.6 Log-Auswertung: Calamaris und Sarg	455
13.6.1 Calamaris	455
13.6.2 Sarg	457
13.7 Unsichtbar: transparent proxy	458
13.8 Ab in den Pool – Verzögerung mit delay_pools	459
13.8.1 Funktionsweise – alles im Eimer!	459
13.8.2 Details – Klassen, Eimer und ACLs richtig wählen	460
13.9 Familienbetrieb: Sibling, Parent und Co.	462
13.9.1 Grundlagen	463
13.9.2 Eltern definieren	464
13.9.3 Geschwister definieren	464

13.9.4	Load Balancing	465
13.9.5	Inhalte eigenständig abrufen: always_direct	465
13.10	Cache-Konfiguration	466
13.10.1	Cache-Arten: Hauptspeicher und Festplatten	466
13.10.2	Hauptspeicher-Cache	467
13.10.3	Festplatten-Cache	467
13.10.4	Tuning	470
14	Kerberos	471
14.1	Begriffe im Zusammenhang mit Kerberos	472
14.2	Die Funktionsweise von Kerberos	472
14.3	Installation und Konfiguration des Kerberos-Servers	473
14.3.1	Starten und Stoppen der Dienste	474
14.3.2	Konfiguration der Datei »/etc/krb5.conf«	475
14.3.3	Konfiguration der Datei »kdc.conf«	477
14.4	Initialisierung und Testen des Kerberos-Servers	481
14.4.1	Verwalten der Principals	483
14.5	Kerberos und PAM	487
14.5.1	Konfiguration der PAM-Dateien auf einem openSUSE-System	488
14.5.2	Testen der Anmeldung	488
14.6	Neue Benutzer mit Kerberos-Principal anlegen	489
14.7	Hosts und Dienste	490
14.7.1	Einträge entfernen	493
14.8	Konfiguration des Kerberos-Clients	494
14.8.1	PAM und Kerberos auf dem Client	495
14.9	Replikation des Kerberos-Servers	496
14.9.1	Bekanntmachung aller KDCs im Netz	496
14.9.2	Konfiguration des KDC-Masters	499
14.9.3	Konfiguration des KDC-Slaves	501
14.9.4	Replikation des KDC-Masters auf den KDC-Slave	502
14.10	Kerberos-Policies	504
14.11	Kerberos in LDAP einbinden	507
14.11.1	Konfiguration des LDAP-Servers	508
14.11.2	Zurücksichern der alten Datenbank	517

14.11.3	Erstellung der Service-Keys in der Standard-»keytab«-Datei	520
14.11.4	Bestehende LDAP-Benutzer um Kerberos-Principal erweitern	521
14.12	Neue Benutzer in den LDAP-Baum aufnehmen	526
14.13	Authentifizierung am LDAP-Server über »GSSAPI«	527
14.13.1	Authentifizierung einrichten	527
14.13.2	Den zweiten KDC an den LDAP-Server anbinden	533
14.14	Konfiguration des LAM Pro	533

15 Samba 4 537

15.1	Vorüberlegungen	537
15.2	Konfiguration von Samba 4 als Domaincontroller	538
15.2.1	Das Provisioning	541
15.2.2	Konfiguration des Bind9	542
15.3	Testen des Domaincontrollers	546
15.3.1	Testen des DNS-Servers	548
15.3.2	Test des Verbindungsaufbaus	549
15.3.3	Einrichtung des Zeitservers	551
15.4	Benutzer- und Gruppenverwaltung	552
15.5	Benutzer- und Gruppenverwaltung über die Kommandozeile	553
15.5.1	Verwaltung von Gruppen über die Kommandozeile	553
15.5.2	Verwaltung von Benutzern über die Kommandozeile	558
15.5.3	Setzen der Passwortrichtlinien	562
15.5.4	Passwortrichtlinien mit Password Settings Objects (PSO)	563
15.6	Die Remote Server Administration Tools (RSAT)	564
15.6.1	Die RSAT einrichten	564
15.6.2	Beitritt eines Windows-Clients zur Domäne	565
15.6.3	Einrichten der RSAT	566
15.6.4	Benutzer- und Gruppenverwaltung mit den RSAT	566
15.7	Gruppenrichtlinien	567
15.7.1	Verwaltung der GPOs mit den RSAT	567
15.7.2	Erste Schritte mit der Gruppenrichtlinienverwaltung	568
15.7.3	Eine Gruppenrichtlinie erstellen	569
15.7.4	Die Gruppenrichtlinie mit einer OU verknüpfen	572
15.7.5	GPOs über die Kommandozeile	576

15.8 Linux-Clients in der Domäne	577
15.8.1 Bereitstellen von Freigaben	583
15.8.2 Mounten über »pam_mount«	584
15.8.3 Umstellen des grafischen Logins	587
15.9 Zusätzliche Server in der Domäne	588
15.9.1 Einen Fileserver einrichten	589
15.9.2 Ein zusätzlicher Domaincontroller	594
15.9.3 Konfiguration des zweiten DC	596
15.9.4 Einrichten des Nameservers	596
15.9.5 Testen der Replikation	599
15.9.6 Weitere Tests	601
15.9.7 Einrichten des Zeitserver	601
15.10 Die Replikation der Freigabe »sysvol« einrichten	602
15.10.1 Einrichten des rsync-Servers	602
15.10.2 Einrichten von rsync auf dem PDC-Master	603
15.11 Was geht noch mit Samba 4?	607
16 NFS	609
<hr/>	
16.1 Unterschiede zwischen NFSv3 und NFSv4	609
16.2 Funktionsweise von NFSv4	610
16.3 Einrichten des NFSv4-Servers	611
16.3.1 Konfiguration des Pseudodateisystems	611
16.3.2 Anpassen der Datei »/etc/exports«	612
16.3.3 Tests für den NFS-Server	614
16.4 Konfiguration des NFSv4-Clients	616
16.5 Konfiguration des idmapd	617
16.6 Optimierung von NFSv4	619
16.6.1 Optimierung des NFSv4Servers	619
16.6.2 Optimierung des NFSv4-Clients	620
16.7 NFSv4 und Firewalls	621
16.8 NFS und Kerberos	622
16.8.1 Erstellung der Principals und der keytab-Dateien	622
16.8.2 Kerberos-Authentifizierung unter Debian und Ubuntu	624
16.8.3 Kerberos-Authentifizierung auf openSUSE und CentOS	624

16.8.4	Anpassen der Datei »/etc/exports«	624
16.8.5	Einen NFS-Client für Kerberos unter Debian und Ubuntu konfigurieren .	625
16.8.6	Einen NFS-Client für Kerberos unter openSUSE und CentOS konfigurieren	625
16.8.7	Testen der durch Kerberos abgesicherten NFS-Verbindung	625
16.8.8	Testen der Verbindung	626

17 LDAP 629

17.1	Einige Grundlagen zu LDAP	630
17.1.1	Was ist ein Verzeichnisdienst?	630
17.1.2	Der Einsatz von LDAP im Netzwerk	631
17.1.3	Aufbau des LDAP-Datenmodells	632
17.1.4	Objekte	632
17.1.5	Attribute	633
17.1.6	Das Schema	634
17.1.7	Das LDIF-Format	637
17.2	Zu den hier verwendeten Distributionen	638
17.3	Installation der Symas-Pakete	639
17.3.1	Die zwei Konfigurationsarten	643
17.3.2	Die Datenbank-Backends	644
17.3.3	Grundkonfiguration des LDAP-Servers (statisch)	645
17.3.4	Grundkonfiguration des LDAP-Servers (dynamisch)	646
17.3.5	Anlegen der ersten Objekte	654
17.4	Die Verbindung zum LDAP-Server über TLS absichern	656
17.4.1	Erstellen der Zertifizierungsstelle	656
17.4.2	Erstellen des Serverzertifikats	657
17.4.3	Signieren des Zertifikats	657
17.4.4	Zertifikate in die »slapd.conf« eintragen	658
17.4.5	Zertifikate in die dynamische Konfiguration eintragen	658
17.4.6	Konfiguration des LDAP-Clients	659
17.5	Einrichtung des sssd	660
17.5.1	Anlegen eines Testbenutzers	665
17.6	Grafische Werkzeuge für die LDAP-Verwaltung	666
17.7	Änderungen mit »ldapmodify«	667
17.7.1	Interaktive Änderung mit »ldapmodify«	668
17.7.2	Änderungen über eine LDIF-Datei mit »ldapmodify«	668

17.8	Absichern des LDAP-Baums mit ACLs	669
17.9	Grundlegende ACLs	673
17.10	Der neue LDAP-Admin	676
17.10.1	Anlegen der Objekte	677
17.11	Absichern der Passwörter	678
17.12	ACLs mit regulären Ausdrücken	679
17.12.1	ACLs vor dem Einsatz testen	683
17.13	Filter zur Suche im LDAP-Baum	685
17.13.1	Die Fähigkeiten des LDAP-Servers testen	686
17.13.2	Einfache Filter	687
17.13.3	Filter mit logischen Verknüpfungen	688
17.13.4	Einschränkung der Suchtiefe	689
17.14	Verwendung von Overlays	690
17.14.1	Overlays am Beispiel von »dynlist«	690
17.14.2	Weitere Overlays	694
17.15	Replikation des DIT	696
17.15.1	Vorbereitungen für die Replikation	697
17.15.2	Einrichtung der Replikation	698
17.15.3	Einrichtung einer Multiprovider-Replikation	706
17.16	Weiterleitungen für den Mailserver Postfix	712
17.17	Benutzerauthentifizierung von Dovecot über LDAP	714
17.18	Benutzerauthentifizierung am Proxy Squid über LDAP	717
17.18.1	Die Authentifizierung über LDAP aktivieren	717
17.18.2	Benutzerbezogene Authentifizierung	719
17.18.3	Gruppenbezogene Authentifizierung	719
17.19	Benutzerauthentifizierung am Webserver Apache über LDAP	720
17.19.1	Konfiguration der Cache-Parameter	721
17.19.2	Konfiguration der Zugriffsparameter	722
17.20	Und was geht sonst noch alles mit LDAP?	723
18	Druckserver	725

18.1	CUPS administrieren	726
18.2	Policies	731
18.2.1	Location-Policies	732
18.2.2	Operation Policies	733

18.2.3	Weitere Konfigurationsmöglichkeiten	734
18.2.4	Browsing	736
18.3	Drucker und Klassen einrichten und verwalten	736
18.3.1	Drucker einrichten	737
18.3.2	Klassen einrichten	738
18.4	Druckerquotas	739
18.5	CUPS über die Kommandozeile	740
18.5.1	Einstellen eines Standarddruckers	740
18.5.2	Optionen für einen Drucker verwalten	741
18.6	PPD-Dateien	743
18.7	Noch mehr Druck	744

TEIL IV Infrastruktur

19 Hochverfügbarkeit 747

19.1	Das Beispiel-Setup	747
19.2	Installation	748
19.2.1	Debian 11 und Ubuntu 22.04 LTS	748
19.2.2	CentOS Stream	748
19.2.3	openSUSE Leap	749
19.3	Einfache Vorarbeiten	749
19.4	Shared Storage mit DRBD	749
19.4.1	Grundlegende Konfiguration	750
19.4.2	Die wichtigsten Konfigurationsoptionen	751
19.4.3	Die DRBD-Ressource in Betrieb nehmen	752
19.5	Grundkonfiguration der Clusterkomponenten	755
19.5.1	Pacemaker und Corosync: das Benachrichtigungssystem	755
19.5.2	Pacemaker: der Ressourcenmanager	758
19.5.3	Ein Quorum deaktivieren	760
19.6	Dienste hochverfügbar machen	762
19.6.1	Die erste Ressource: eine hochverfügbare IP-Adresse	763
19.6.2	Hochverfügbarkeit am Beispiel von Apache	766
19.6.3	DRBD integrieren	769
19.6.4	Fencing	773

20	Virtualisierung	775
20.1	Einleitung	775
20.2	Für den Sysadmin	776
20.3	Servervirtualisierung	780
20.3.1	KVM	781
20.3.2	Xen	783
20.4	Netzwerkgrundlagen	784
20.5	Management und Installation	785
20.5.1	Einheitlich arbeiten: »libvirt«	786
20.5.2	Konsolenbasiertes Management: virsh	789
20.5.3	Virtuelle Maschinen installieren	792
20.5.4	virt-install	794
20.5.5	Alleskönner: Der Virtual Machine Manager	797
20.5.6	Zusätzliche Konsolentools	801
20.6	Umzugsunternehmen: Live Migration	802
20.6.1	Vorbereitungen	803
20.6.2	Konfiguration im Virtual Machine Manager	803
21	Containervirtualisierung mit Docker und Podman	805
21.1	Einführung, Installation und Grundlagen für den Betrieb	805
21.1.1	Was ist ein Container?	805
21.1.2	Container vs. VM	806
21.1.3	Entstehung und Geschichte	806
21.1.4	Versionen	807
21.1.5	Docker oder Podman?	808
21.1.6	Installation von Docker	809
21.1.7	Installation von Podman	811
21.1.8	Ergänzungen zur Installation, erster Systemtest	811
21.1.9	Betrieb hinter einem Proxy	813
21.1.10	Konfiguration der Laufzeitumgebung	814
21.2	Management von Images und Containern	815
21.2.1	Etwas Terminologie	815
21.2.2	Das Command Line Interface	816
21.2.3	Erste Schritte: hello-world	817

21.2.4	Löschen von Containern und Images	818
21.2.5	Image-Namen, Docker Hub und weitere Registrys	819
21.2.6	Handling von Containern	820
21.2.7	Prozessverwaltung	822
21.2.8	Umgebungsvariablen	823
21.2.9	Logging	824
21.2.10	Verteilung von Images über Dateiversand	825
21.2.11	Ausgaben filtern und/oder formatieren	825
21.2.12	Restart-Policies: Verhalten beim Host-Restart	827
21.2.13	Container limitieren	828
21.2.14	Packungsdichte	831
21.2.15	Systeminformationen und Aufräumarbeiten	831
21.3	Docker-Networking	832
21.3.1	User Defined Networks	833
21.3.2	Portmapping	834
21.3.3	»/etc/hosts«-Einträge beim Containerstart	835
21.4	Containerdaten und Persistenz	836
21.4.1	Aufbau von Images und Containern	836
21.4.2	Bind Mounts und Volumes	837
21.4.3	Weitere Möglichkeiten	840
21.4.4	Informationsbeschaffung	840
21.5	Erstellen eigener Images mit Dockerfiles	842
21.5.1	Einfaches Comitten von Anpassungen	842
21.5.2	Dockerfiles und »docker build«: Basics	844
21.5.3	Der Build-Cache und »docker build --pull«	844
21.5.4	Dangling Images	845
21.5.5	Die Dockerfile-Direktiven: Ein Überblick	846
21.5.6	Ein komplexeres Beispiel mit ENV, COPY und CMD	847
21.5.7	CMD und/oder ENTRYPOINT	848
21.5.8	Verwendung eigener Entrypoint-Skripte	850
21.5.9	».dockerignore«-Files	851
21.5.10	Healthchecks	851
21.5.11	Multistage-Builds	853
21.5.12	Best Practices	854
21.6	Multi-Container-Rollout mit Docker Compose	855
21.6.1	Installation	855
21.6.2	Basics	856
21.6.3	Ein erstes Beispiel	857
21.6.4	Build and Run	858

21.6.5	Environment und Portmappings	859
21.6.6	Volumes in Compose	860
21.6.7	Flexible Compose-Konfigurationen durch Umgebungsvariablen	861
21.6.8	Noch mal Restart-Policies	862
21.7	Betrieb und Verwendung einer eigenen Registry	862
21.7.1	Vorbereitungen in einer (virtuellen) Test-/Schulungsumgebung	863
21.7.2	Heute mal kein TLS/HTTPS	864
21.7.3	Harbor	866
21.7.4	Docker Registry	867
21.7.5	Arbeiten mit einer privaten Registry	869

TEIL V Kommunikation

22 Netzwerk 873

22.1	Vorwort zu Predictable Network Interface Names	873
22.2	Netzwerkkonfiguration mit iproute2	874
22.2.1	Erste Schritte	874
22.2.2	Die Syntax von ip	877
22.2.3	Links ansehen und manipulieren: ip link	877
22.2.4	IP-Adressen ansehen und manipulieren: ip address	879
22.2.5	Manipulation von ARP-Einträgen: ip neighbour	883
22.3	Routing mit ip	885
22.3.1	Routing-Informationen anzeigen	885
22.3.2	Da geht noch mehr: »Advanced Routing«	887
22.3.3	Die vorhandenen Regeln ansehen	888
22.3.4	Eine neue Routing-Tabelle anlegen	889
22.3.5	Ändern der Policy Routing Database	889
22.3.6	Routing über mehrere Uplinks	891
22.3.7	Fazit bis hierher	896
22.4	Bonding	896
22.4.1	Bonding-Konfiguration	897
22.4.2	Bonding unter Debian	900
22.4.3	Bonding unter Ubuntu	900
22.4.4	Bonding unter CentOS	901
22.4.5	Bonding unter openSUSE Leap	902

22.5 IPv6	902
22.5.1 Die Vorteile von IPv6	904
22.5.2 Notation von IPv6-Adressen	904
22.5.3 Die Netzmasken	905
22.5.4 Die verschiedenen IPv6-Adressarten	905
22.5.5 Es geht auch ohne ARP	907
22.5.6 Feste Header-Länge	908
22.5.7 IPv6 in der Praxis	910
22.6 Firewalls mit netfilter und iptables	911
22.6.1 Der Weg ist das Ziel – wie Pakete durch den Kernel laufen	912
22.6.2 Einführung in iptables	913
22.6.3 Regeln definieren	915
22.6.4 Die klassischen Targets	917
22.6.5 Ein erster Testlauf	917
22.6.6 Rein wie raus: Stateful Packet Inspection	918
22.6.7 Das erste Firewallskript	920
22.6.8 Externe Firewall	922
22.6.9 Logging	928
22.6.10 Network Address Translation und Masquerading	930
22.6.11 Weitere nützliche Module für iptables	931
22.6.12 Abschlussbemerkung	934
22.7 DHCP	934
22.7.1 Funktionsweise	934
22.7.2 Konfiguration	935
23 DNS-Server	939
<hr/>	
23.1 Funktionsweise	939
23.1.1 Unterschied: rekursiv und autoritativ	941
23.1.2 Einträge im DNS: Resource Records	941
23.1.3 Die Grundkonfiguration	942
23.1.4 Zonendefinitionen	944
23.1.5 Die erste vollständige Zone	949
23.1.6 Die hint-Zone	950
23.1.7 Reverse Lookup	952
23.1.8 Secondary-Server	954
23.1.9 DNS-Server und IPv6	956

23.2	Vertrauen schaffen mit DNSSEC	957
23.2.1	Die Theorie: »Wie arbeitet DNSSEC?«	957
23.2.2	Anpassungen am Server	959
23.2.3	Schlüssel erzeugen	960
23.2.4	Schlüssel der Zone hinzufügen und die Zone signieren	961
23.2.5	Signierte Zone aktivieren	963
23.2.6	Signierung prüfen	963
23.2.7	Die Signierung veröffentlichen	965
23.2.8	Weniger anstrengend: Mehr Automatismus!	966
23.2.9	Fazit	967
23.3	Client-Anfragen absichern mit »DNS over HTTPS (DoH)«	967
23.3.1	Installation	967
23.3.2	Vorbereitungen	968
23.3.3	Konfiguration	969
23.3.4	Funktionstest	970
23.3.5	Client-Konfiguration	971

24 OpenSSH 973

24.1	Die SSH-Familie	973
24.1.1	Die Clients: ssh, scp, sftp	974
24.1.2	Der Server: sshd	976
24.2	Schlüssel statt Passwort	978
24.2.1	Schlüssel erzeugen	978
24.2.2	Passwortloses Login	979
24.2.3	Der SSH-Agent merkt sich Passphrasen	980
24.3	X11-Forwarding	981
24.4	Portweiterleitung und Tunneling	982
24.4.1	SshFS: Entfernte Verzeichnisse lokal einbinden	983

25 Administrationstools 985

25.1	Was kann dies und jenes noch?	985
25.1.1	Der Rsync-Daemon	985
25.1.2	Wenn's mal wieder später wird: screen	987

25.1.3	Anklopfen mit nmap	987
25.1.4	Netzwerkinspektion: netstat	991
25.1.5	Zugreifende Prozesse finden: lsof	993
25.1.6	Was macht mein System? top	997
25.1.7	Wenn gar nichts mehr geht – Debugging mit strace	1001
25.1.8	Prüfung der Erreichbarkeit mit my traceroute	1006
25.1.9	Subnetzberechnung mit ipcalc	1007
25.2	Aus der Ferne – Remote-Administrationstools	1008
25.2.1	PuTTY	1009
25.2.2	WinSCP	1012
25.2.3	Synergy	1013
25.2.4	Eine für immer: mosh	1015

26 Versionskontrolle 1017

26.1	Philosophien	1018
26.1.1	Lokal	1018
26.1.2	Zentral	1019
26.1.3	Dezentral	1020
26.2	Versionskontrollsysteme	1020
26.2.1	CVS	1021
26.2.2	Apache Subversion	1024
26.2.3	GNU Bazaar	1026
26.2.4	Mercurial	1028
26.2.5	Git	1030
26.3	Kommandos	1032
26.4	Serverdienste	1033
26.4.1	Git-Server mit Gitolite	1033
26.4.2	Git-Server mit Gitea	1037

TEIL VI Automatisierung

27 Scripting	1043
27.1 Aufgebohrte Muscheln	1043
27.2 Vom Suchen und Finden: ein kurzer Überblick	1044
27.2.1 Die Detektive: grep, sed und awk	1044
27.2.2 Reguläre Ausdrücke verstehen und anwenden	1045
27.3 Fortgeschrittene Shell-Programmierung	1048
27.3.1 Expansionsschemata	1048
27.3.2 Umgebungsvariablen	1052
27.3.3 »Back to bash«: ein tieferer Blick in die Muschel	1053
27.3.4 Logging in Skripten	1057
27.4 Tipps und Tricks aus der Praxis	1060
27.4.1 Aufräumkommando	1061
27.4.2 IFS	1061
27.4.3 Datumsmagie	1062
27.4.4 E-Mails aus einem Skript versenden	1062
27.4.5 Interaktive Programme steuern	1063
28 Konfigurationsmanagement mit Ansible	1065
28.1 Einführung und Installation	1065
28.1.1 Was ist Ansible?	1065
28.1.2 Geschichte und Versionen	1067
28.1.3 Setup/Laborumgebung	1067
28.1.4 Ansible-Installation auf dem Control Host	1068
28.1.5 Authentifizierung und Autorisierung auf den Target Hosts	1071
28.1.6 Einrichten der SSH-Public-Key-Authentifizierung	1072
28.1.7 Ein Ad-hoc-Test ohne jegliche Konfiguration	1072
28.2 Basiseinrichtung und erstes Inventory-Management	1074
28.2.1 Verzeichnisstruktur einrichten	1074
28.2.2 Grundkonfiguration (»ansible.cfg«)	1075
28.2.3 Erstellen und Verwalten eines statischen Inventorys	1076
28.2.4 Inventory-Aliasse und Namensbereiche	1078
28.2.5 Jenseits von Ping	1079
28.2.6 Ein etwas komplexeres Beispiel	1081
28.2.7 Alternative bzw. mehrere Inventorys	1082

28.3	Ad-hoc-Kommandos und Patterns	1084
28.3.1	Ad-hoc-Kommandos	1084
28.3.2	Use Cases jenseits von »command« und »shell«	1085
28.3.3	Idempotenz	1086
28.3.4	Interne Funktionsweise	1086
28.3.5	Die Ansible-Konsole	1088
28.3.6	Patterns zum Adressieren von Hosts	1089
28.4	Die Konfigurations- und Serialisierungssprache YAML	1090
28.4.1	Syntax und Struktur	1090
28.4.2	YAML-Files editieren	1091
28.4.3	Listen und Maps	1092
28.4.4	Verschachtelte Strukturen	1093
28.4.5	Textpassagen und Block-Ausdrücke	1094
28.4.6	Das Nichts in YAML	1095
28.5	Playbooks und Tasks: die Grundlagen	1095
28.5.1	Hallo Ansible – das allererste Playbook	1096
28.5.2	Formulierung von Tasks	1099
28.5.3	Beenden von Plays	1100
28.5.4	Fehlerbehandlung, Retry-Files	1101
28.5.5	Tags	1102
28.5.6	Das Kommando »ansible-playbook«	1103
28.5.7	Eine exemplarische Apache-Installation	1104
28.5.8	Handler: Tasks nur bei Changes durchführen	1108
28.6	Playbooks und Tasks: fortgeschrittene Methoden	1112
28.6.1	Variablen	1112
28.6.2	Registrierte Variablen	1118
28.6.3	Facts und implizite Variablen	1122
28.6.4	Bedingte Ausführung mit »when«	1124
28.6.5	Jinja und Templates	1125
28.6.6	Schleifen	1128
28.6.7	Fehlerbehandlung mit »failed_when« und »ignore_errors«	1133
28.6.8	Blocks	1134
28.6.9	Lookup-Plug-ins	1134
28.6.10	Umgebungsvariablen setzen	1136
28.7	Module und Collections verwenden	1137
28.7.1	Collections	1137
28.7.2	Module	1141
28.7.3	Module zur Kommandoausführung	1142
28.7.4	Module zur Paketverwaltung	1143

28.7.5	Module zur Verwaltung von Dateien und Dateiinhalten	1145
28.7.6	Module für weitere typische Verwaltungsaufgaben	1148
28.7.7	Spezialmodule (Kontrollflusssteuerung etc.)	1151
28.8	Nächste Schritte	1153
29	Monitoring – wissen, was läuft	1155
<hr/>		
29.1	Monitoring mit Checkmk	1155
29.2	Installation der Pakete	1155
29.2.1	Installation von Checkmk unter openSUSE	1156
29.2.2	Installation von Checkmk unter Debian/Ubuntu	1156
29.2.3	Installation von Checkmk unter CentOS	1156
29.2.4	Die erste Kontrolle – klappt alles?	1156
29.3	Einrichtung der ersten Monitoring-Instanz	1157
29.4	Server, Geräte und Dienste überwachen	1160
29.5	Installation des Checkmk-Agenten	1161
29.6	Anlegen eines Hosts	1162
29.7	Betriebs- und Fehlerzustände von Host und Services im Überblick	1163
29.8	Konfiguration durch Regelsätze	1164
29.8.1	Arbeiten in Host-Ordnern	1165
29.8.2	Keine Alarme für Testsysteme	1167
29.8.3	Unterschiedliche Alarmschwellen bei Dateisystemen	1168
29.8.4	Service Discovery Rules: Gezielt Prozesse überwachen	1170
29.8.5	HTTP, TCP und E-Mail: Netzwerkdienste überwachen	1172
29.9	Notifications	1173
29.9.1	Anlegen weiterer Kontaktgruppen	1173
29.9.2	Test der E-Mail-Zustellung	1174
29.9.3	Alarmierung per SMS	1174
29.9.4	Wann wird ein Fehler zum HARD STATE?	1175
29.9.5	Definieren von Notification Periods	1176
29.10	Alarme managen	1176
29.10.1	Die mächtige Suche von Checkmk	1178
29.11	Weitere Fähigkeiten von Checkmk	1179
29.12	Fazit	1180

TEIL VII Sicherheit, Verschlüsselung und Zertifikate

30	Sicherheit	1183
30.1	Weniger ist mehr	1184
30.2	chroot	1184
30.2.1	Dienste	1185
30.3	Selbstabsicherung: AppArmor	1187
30.3.1	Status und Betriebsarten	1188
30.3.2	Eigene Profile erstellen	1190
30.4	Gotcha! Intrusion-Detection-Systeme	1193
30.4.1	snort und Co.	1194
30.5	Installation und Konfiguration	1195
30.5.1	Vorbereitungen	1196
30.5.2	Kompilieren und installieren	1197
30.5.3	Basiskonfiguration	1198
30.5.4	Ein erster Test: ICMP	1199
30.5.5	Start-Skript erstellen: systemd	1200
30.6	Immer das Neueste vom Neuen: pulledpork	1201
30.7	Klein, aber oho: fail2ban	1204
30.7.1	Konfiguration	1204
30.7.2	Aktive Sperrungen	1207
30.7.3	Reguläre Ausdrücke	1209
30.8	OpenVPN	1210
30.8.1	Serverinstallation – OpenVPN, PKI und Co.	1211
30.8.2	CentOS/openSUSE Leap: easy-rsa	1215
30.8.3	Gemeinsam weiter	1218
30.8.4	Für den Roadwarrior	1220
30.8.5	Start-Skript?	1222
30.8.6	Site-to-site	1226
30.8.7	Simple-HA	1228
30.8.8	Tipps und Tricks	1229
30.9	Schnell, Modern, Sicher: WireGuard	1232
30.9.1	Schnell einen Tunnel einrichten	1233
30.9.2	Die dunkle Seite des Mondes	1235
30.9.3	Dauerhafte Tunnel mit »systemd«	1235

30.9.4	Alle machen mit: »Hub and Spoke«	1237
30.9.5	Tipps und Tricks	1238
30.10	Fazit	1239
31	Verschlüsselung und Zertifikate	1241
31.1	Definition und Historie	1241
31.2	Moderne Kryptologie	1243
31.2.1	Symmetrische Verschlüsselung	1243
31.2.2	Asymmetrische Verschlüsselung	1244
31.3	Den Durchblick behalten	1245
31.3.1	Das Grundproblem	1245
31.3.2	Verwendungszwecke	1246
31.3.3	Umsetzung mithilfe einer PKI	1246
31.3.4	X.509	1247
31.3.5	Ein anderer Ansatz: PGP (Web-of-Trust)	1249
31.4	Einmal mit allem und kostenlos bitte: Let's Encrypt	1249
31.4.1	Wie funktioniert das?	1250
31.4.2	Einschränkungen	1251
31.4.3	Der Client certbot	1251
31.5	In der Praxis	1253
31.5.1	Einrichtung einer PKI mit Server- und E-Mail-Zertifikaten	1253
31.5.2	Lokale Zertifikatsausstellung wie Let's Encrypt: acme2certifier	1264
31.5.3	E-Mail-Verschlüsselung	1271
31.6	Neben der Kommunikation – Dateiverschlüsselung	1279
31.6.1	Dateien	1279
31.6.2	Devices	1280
31.6.3	Festplatten/System	1282
	Die Autoren	1287
	Index	1289

Vorwort

Willkommen zur siebten Auflage von »Linux-Server. Das umfassende Handbuch«! Auch mehr als 10 Jahre nach der Erstaufgabe finden sich noch neue Themen oder große Änderungen, die eine neue Auflage füllen können. Das gilt nicht nur für die Cloud! Auch der eigene Server, den Sie in Ihrem Rechenzentrum oder Serverraum pflegen, ist wichtiger denn je. Innovationen und Änderungen in diesem Bereich werden uns noch lange begleiten.

Wieder standen wir vor der Wahl, welche Distributionen wir für diese Ausgabe bearbeiten sollten. Debian und Ubuntu waren von Anfang an fest eingeplant. Auch Suse war fest gesetzt. Bei Redhat haben wir uns dafür entschieden, wieder auf die Stream-Variante zu setzen, die die Zukunft der neuen Ableger AlmaLinux oder Rocky Linux nicht ganz klar ist. Wir werden diese Situation aber weiterhin beobachten, denn es ist davon auszugehen, dass sich in diesem Bereich in den nächsten Jahren viel tun wird.

In dieser Aktualisierung gab es wieder viele kleine und große Änderungen. Am wichtigsten ist die neue Version von OpenLDAP. 14 Jahre nachdem Version 2.4 erschienen ist, wurde mit Version 2.6 eine vollständige Überarbeitung veröffentlicht, auf die wir ausführlich in Kapitel 17 eingehen. Beim Thema Datenbanken haben wir uns schon in der letzten Auflage dafür entschieden, MySQL durch MariaDB zu ersetzen, da MySQL teilweise gar nicht mehr in den Repositories der Distributionen vorhanden ist. Wie wir heute sagen können: Eine gute Entscheidung. Bei den Distributionen hat sich – wie bereits angeführt – auch etwas geändert. Bei Debian ist die Version Debian Bullseye neu hinzugekommen, bei Suse sind wir auf Suse Leap 15.4 umgestiegen. Von Ubuntu ist eine neue LTS-Version auf dem Markt, die Version 22.04. Bei CentOS werden wir weiterhin die Version Stream nutzen.

Fast alle Kapitel wurden von uns komplett überarbeitet, teilweise neu geschrieben, um möglichst aktuell zu bleiben. Hier war es uns wieder besonders wichtig, die neuen Funktionen von Programmen aufzunehmen und eventuell ältere Optionen und Vorgehensweisen herauszunehmen. Zudem haben wir auch wieder viele Anregungen und Kommentare erhalten, die uns dazu inspiriert haben, eine neue Auflage zu schreiben. Außerdem schreiben ein paar der Autoren noch Bücher zu speziellen Themen und haben daraus die eine oder andere Idee übernommen. Sicherlich hätten wir an manchen Stellen noch tiefer einsteigen können, aber der Umfang des Buches kommt so langsam an die Grenzen des technisch Machbaren. Zudem soll das Buch einen Überblick über möglichst viele verschiedene Dienste geben; es kann und will kein Wälzer sein, der alle Dienste bis zum letzten Bit beschreibt.

Wie schon bei der sechsten Auflage wollen wir Ihnen mit diesem Buch eine Anleitung bieten, wie Sie die verschiedensten Dienste, die Ihnen ein Linux-System bereitstellen kann, schnell und einfach konfigurieren. Ohne große Umwege geht es über die Konfiguration hin zu einem funktionsfähigen Dienst, den Sie dann an Ihre eigenen Bedürfnisse anpassen können.

Zudem haben wir alle großen Neuerungen für Sie so zusammengefasst, dass Sie auch neue Techniken nachlesen und umsetzen können.

Wir wollen Ihnen ein Nachschlagewerk an die Hand geben, das Sie mit vielen verschiedenen Techniken und Diensten vertraut macht. In den einzelnen Kapiteln gehen wir auch immer wieder auf Besonderheiten der verschiedenen Distributionen ein. Gerade durch die Vielfalt der Dienste und Techniken können Sie dieses Buch wie ein Schweizer Taschenmesser nutzen: immer griffbereit und immer das richtige Werkzeug dabei. Mit jeder Auflage bekommt dieses Schweizer Taschenmesser ein paar Werkzeuge mehr, und die bestehenden Werkzeuge wurden an vielen Stellen noch schärfer und präziser gemacht. Auch in dieser Auflage haben wir viele Beispiele aus unserer täglichen Arbeit aufgenommen, denn das, was wir in den verschiedenen Unternehmen erleben und einrichten, ist immer eine gute Grundlage, um Ihnen zu helfen, möglichst schnell ans Ziel zu gelangen.

Für wen haben wir das Buch geschrieben?

Dieses Buch richtet sich an alle Linux-Systemadministratoren, die immer wieder vor der Aufgabe stehen, neue Dienste in ihrem Netzwerk zu etablieren, und die am Anfang einen möglichst schnellen und kompakten Einstieg in das Thema wünschen. Grundlegende Linux-Kenntnisse, wie sie zum Beispiel in LPIC-1 verlangt werden, sollten auf jeden Fall schon vorhanden sein, damit Sie die einzelnen Dienste erfolgreich in das eigene Netz integrieren können.

Wie können Sie mit diesem Buch arbeiten?

Wir haben das Buch so geschrieben, dass Sie gezielt mit den Beispielen aus den einzelnen Kapiteln einen neuen Dienst konfigurieren und testen können. An vielen Stellen verweisen wir aber auch auf andere Dienste, die hier im Buch beschrieben sind, um Ihnen die Möglichkeit zu geben, auch komplexere Aufgaben zu realisieren.

Was dieses Buch nicht ist

Dieses Buch ist kein Lehrbuch, um den Umgang mit Linux von Grund auf zu verstehen, dafür gibt es viele andere Bücher auf dem Markt. Auch war das Buch von Anfang an nicht dazu gedacht, einen oder mehrere einzelne Dienste bis ins Letzte zu konfigurieren. Denken Sie an Ihr Schweizer Taschenmesser: Es kann Ihnen bei vielen Aufgaben helfen, aber für spezielle Aufgaben gibt es spezielle Werkzeuge. Das Gleiche gilt für unser Buch.

Viele Aufgaben können Sie mithilfe unseres Buches erledigen, aber wenn es dann sehr speziell wird, brauchen Sie ein Buch, das genau dieses eine Thema bis in kleinste Detail beschreibt.

Über 10 Jahre Linux-Server Buch

Am 28. Januar 2011 wurde die erste Auflage unseres Buches Linux-Server veröffentlicht.

Die Idee zum Buch erblickte im April 2008 das Licht der Welt, bis zum Erscheinen der ersten Auflage vergingen also fast drei Jahre. Nach dem Ausstieg des Ideengebers Marcus Fischer fand sich ein erstes Autorenteam, das sich im Januar 2009 mit allen Beteiligten in einem Café in Stuttgart traf, um die weitere Vorgehensweise zu besprechen. Leider hat sich dieses erste Team bis auf Stefan Kania und Dirk Deimeke wieder zerschlagen, neue Mitautoren wurden gesucht und mit Charly Kühnast, Daniel van Soest und Stefan Semmelroggen auch gefunden. Im November fand ein Treffen des neuen Teams auf der allerersten OpenRheinRuhr-Konferenz im Bottroper Saalbau statt, ein Mitglied des Teams konnte damals leider nur telefonisch teilnehmen. Nachdem die Eckbedingungen für das Buch klar waren, fand die weitere Kommunikation überwiegend auf der eigens dafür eingerichteten Mailingliste statt. In unregelmäßigen Abständen gab es weitere Audio- und Videokonferenzen, um strittige Punkte und Inhalte miteinander abzustimmen.

Wir durften gemeinsam erleben, wie schwierig es ist, in einem verteilten Team zu arbeiten, bei dem sich die Teilnehmer kaum persönlich kennen. Alle von uns haben verschiedene Hintergründe und Lebensläufe, aber wir sind alle Experten für die Bereiche, die wir im Buch übernommen haben. Viel wurde darüber diskutiert, was in das Buch kommt und wie umfangreich die einzelnen Kapitel werden sollen. Jeder hatte seine Ideen, und wie das so ist: Wenn viele Köche an einem Menü arbeiten, ist für jeden sein Teil der wichtigste. Nachdem wir uns auf die Inhalte geeinigt hatten, konnten wir endlich loslegen. Für die meisten von uns war das Schreiben eine komplett neue Erfahrung. Schulungsunterlagen hatte der eine oder andere schon erstellt, aber ein Buch! Das ist noch mal eine ganz anderer Herausforderung.

Für einige von uns war der Umgang mit \LaTeX neu, aber selbst der größte Skeptiker hat es zum Schluss lieben gelernt. Technisch haben wir anfänglich alles mit Subversion als Versionskontrollsystem versioniert und sind im Laufe der Zeit auf Git umgestiegen.

Es hat geklappt, wie Sie sehen. Über die Jahre wurden wir immer sicherer, und der Inhalt wurde von Auflage zu Auflage besser. Aber wenn man dann so dabei ist, bekommt man immer neue Ideen, was noch alles wichtig sein kann. Jetzt sind wir an die Grenzen des technisch Machbaren gelangt, was den Druck angeht, und mehr als das, was Sie jetzt in der Hand halten, geht nicht. Obwohl wir noch einige Ideen hätten ...

Über die verschiedenen Auflagen hinweg haben wir den einen oder anderen Autor verloren, aber auch immer wieder neue gute Autoren dazu gewonnen. Bei über 10. Jahren sollte man, denken wir, alle Namen nennen, die geholfen haben das Buch zu gestalten. Hier die Liste aller Autoren:

- ▶ Dirk Deimeke
- ▶ Stefan Kania
- ▶ Charly Kühnast
- ▶ Stefan Semmelroggen

- ▶ Daniel van Soest
- ▶ Peer Heinlein
- ▶ Axel Miesen

Auch der verantwortliche Lektor aus dem Verlag hat während der Zeit einige Male gewechselt. Ohne einen guten Lektor kann so ein Projekt wie dieses Buch nicht über so lange Zeit erfolgreich sein. Aus diesem Grund möchten wir auch die Lektoren auflisten, die uns in dieser Zeit mit Rat und Tat zur Seite gestanden haben:

- ▶ Jan Watermann
- ▶ Sebastian Kestel
- ▶ Anne Scheibe
- ▶ Christoph Meister

Wir hatten im Vorfeld nicht gedacht, wie wichtig sie sind, aber wir möchten auch ganz besonders den beiden Korrektorinnen danken, die uns zwischenzeitlich in den Wahnsinn getrieben haben, weil sie mit großer Sorgfalt unsere Fehler identifiziert und angemahnt haben. Sie tragen mit ihrer großartigen Arbeit zum Erfolg dieses Buchs bei:

- ▶ Friederike Daenecke
- ▶ Angelika Glock

Eine weitere Person aus dem Verlag soll hier auch nicht unerwähnt bleiben. Wir möchten unserem Hersteller danken, der bei allen Auflagen für uns als Ansprechpartner für technische Probleme bereitstand und uns wichtige Tipps zum Satz und zu der Bearbeitung der Bilder gegeben hat:

- ▶ Norbert Englert

Nur alle zusammen konnten wir das Buch über die Jahre immer wieder aktualisieren.

Ein ganz großer Dank geht natürlich auch an Sie, denn ohne die Leser kann das beste Buch nicht mehr als 10 Jahre bestehen. Für uns war auch das Feedback der Leserinnen und Leser immer wieder wichtig, denn manchmal bekommt man durch ein Feedback eine ganz andere Sichtweise auf bestimmte Dinge und kann diese Erkenntnis in einer neuen Auflage einfließen lassen.

Haben Sie viel Spaß mit dem Buch und möge es Ihnen bei dem einen oder anderen Projekt gute Hilfe leisten.

Vorwort von Dirk Deimeke

Im April 2008 kam Marcus Fischer, der Autor zahlreicher Ubuntu-Bücher beim Rheinwerk Verlag, mit der Idee auf mich zu, ein Linux-Adminbuch zu schreiben. Da es kein deutsches Werk gibt, das die Lücke zwischen Einsteigerbüchern und Fachbüchern schließt, die sich ei-

nem einzelnen Thema widmen, war und bin ich immer noch Feuer und Flamme. In den folgenden fünf Monaten arbeiteten wir zusammen mit Jan Watermann, dem damaligen Lektor, an dem Konzept des Buches. Uns war zu jedem Zeitpunkt klar, dass es ein Buch werden sollte, das viel Bezug zur Praxis hat und einige Probleme behandelt, denen Linux-Systemadministratoren täglich begegnen. Das schreibt sich so leicht in ein oder zwei Sätzen, aber es war ein längerer Dialog, da jeder eine etwas andere Vorstellung davon hatte, wie das Buch aussehen sollte. Der Begriff »Kochbuch« durfte aufgrund von Markenrechten nicht verwendet werden, traf aber das, was wir machen wollten, am besten.

Nachdem Marcus aufgrund seiner Dissertation keine Zeit mehr hatte, an dem Buch zu arbeiten, ging die Suche nach Autoren los, und Mitstreiter wurden gefunden. Aufgrund interner Schwierigkeiten trennte sich die initiale Gruppe jedoch wieder, und es drohte das Aus. In einem zweiten Anlauf fanden sich dann die Autoren zusammen, die die erste Auflage des Buchs geschrieben haben. Stefan Kania ist außer mir aus der ersten Gruppe dageblieben. Zu uns gestoßen sind für die zweite Auflage Stefan Semmelroggen, Daniel van Soest und Charly Kühnast. Mit der dritten Auflage hat sich das Team leider wieder geändert: Stefan Semmelroggen verließ das Team, und Peer Heinlein stieß dazu. In der vierten Auflage verließ uns Charly Kühnast. In der fünften Auflage dürfen wir jetzt Axel Miesen als zusätzlichen Autor begrüßen, der die Kapitel zu Ansible und Docker beigesteuert hat. Gleichzeitig durften wir für diese Auflage mit einem neuen Lektor – Christoph Meister – zusammenarbeiten.

Anfang 2011 erschien die Ursprungsversion des Buches. Aufgrund von Änderungen in den Distributionen und von Anregungen unserer Leser gingen wir in die zweite Runde für Mitte 2012. Nach den größeren Änderungen der dritten Auflage legten wir mit der vierten Auflage noch eins drauf und haben den Verlag gewechselt – nein, im Ernst, in diese Zeit fiel auch die Umbenennung des Verlags von Galileo Press in Rheinwerk Verlag. Da wir immer noch sehr viele Ideen haben, müssen wir uns neuen Herausforderungen stellen, und zwar, das Format zu halten und nicht zu ausschweifend zu werden. Wir kommen leider sehr nah an die technischen Limits, die ein Buch mit rund 1300 Seiten erreicht.

Und – WOW! – jetzt halten Sie die siebte Auflage in den Händen.

Seit der vierten Auflage bieten wir Unterstützung für die am weitesten verbreiteten Distributionen mit längerer Laufzeit und sind mit openSUSE Leap kompatibel mit dem SUSE Linux Enterprise Server, und mit CentOS sind wir kompatibel mit Red Hat Enterprise Linux. Sie, liebe Leser und Leserinnen, haben diese Änderungen angenommen und uns bewiesen, dass wir auf dem richtigen Weg sind.

Neben einem intensiven Review und einem Test, ob die angegebenen URLs noch funktionieren, habe ich noch kleinere Änderungen in die siebte Auflage aufgenommen.

Natürlich wurden alle Beispiele mit den neuen Versionen der Distributionen getestet und entsprechend angepasst. Wir hoffen, dass wir Ihnen mit diesem Buch weiterhin Unterstützung bei Ihrer täglichen Arbeit geben können!

Danksagung

Allen voran möchte ich meiner Frau danken, ohne die mein Teil an diesem Buch nie möglich gewesen wäre. Christoph Meister vom Rheinwerk Verlag danke ich für seine wertvollen Hinweise und für seine Geduld. Die Zusammenarbeit mit Dir macht Spaß, gerade auch weil Du sehr viel Verständnis für uns »Autoren im Nebenberuf« aufbringst.

Aber auch hinter den Kulissen haben wir bei »den Rheinwerkern« helfende Hände gefunden, allen voran möchte ich unserer Korrektorin Friederike Daenecke danken, die jeden noch so kleinen sprachlichen Fehler gefunden und behoben hat. Danke! Unserem Hersteller Norbert Englert möchte ich danken, weil wir mit seiner Hilfe aus dem Manuskript ein ansehnliches Buch machen konnten.

Mein besonderer Dank gilt aber meinen Mitautoren Daniel, Peer, Axel und Stefan für die tolle Zusammenarbeit.

Dass die Idee, die diesem Buch zugrunde liegt, so erfolgreich ist, damit haben wir nicht gerechnet. Noch viel weniger haben wir damit gerechnet, dass dieses Buch begonnen hat, sich als Standardwerk zu etablieren.

Jetzt halten Sie, liebe Leserin, lieber Leser, bereits die siebte Auflage in Ihren Händen. Sie ist möglich geworden, weil Sie uns mit Ihren Anregungen und Ihrer konstruktiven Kritik motiviert haben. Danke!

Vorwort von Stefan Kania

Bei dieser Auflage lagen meine Schwerpunkte beim Thema OpenLDAP, denn dort hat es die meisten Änderungen gegeben. Nicht nur Kleinigkeiten, sondern nach 14 Jahren wurde wieder eine neue Version veröffentlicht. Im ersten Moment sah es so aus, als würde alles beim Alten bleiben, denn die Konfiguration eines alten OpenLDAP 2.4 konnte direkt übernommen werden. Aber jeder weitere Blick zeigte, wie viel Potential in der neuen Version liegt. Die komplette Replikation wurde erneuert, verbessert und noch performanter gestaltet. Zusätzliche Overlays sorgen für noch mehr Möglichkeiten. Alle Neuerungen kann man in einem Kapitel mit 100 Seiten gar nicht abdecken, aber ich denke, dass Sie einen guten Überblick über die Neuerungen erhalten.

Im Samba-Kapitel habe ich das Thema Dateisystemrechte überarbeitet und einige Dinge so genau beschrieben, wie es auf wenigen Seiten möglich war. Auch die Gruppenrichtlinien für Linux-Clients hätte ich gerne aufgenommen, aber leider reicht der Platz hierfür nicht aus.

Das Kerberos-Kapitel habe ich auf die neue OpenLDAP-Version angepasst und auch hier alles noch mal überarbeitet.

Danksagung

Mein Dank geht dieses Mal in erster Linie an den Verlag, der unser Projekt schon so lange unterstützt und uns die Möglichkeit gibt, soviel Wissen zusammen zu stellen. Bei den Autoren muss ich doch dieses Mal einen meiner Mitstreiter besonders hervorheben, der schon seit ein paar Auflagen den gesamten Satz übernimmt und dafür sorgt, dass das Buch am Ende nicht nur viel Inhalt hat, sondern auch noch gut aussieht. Danke, Daniel, für die viele Arbeit, die du immer wieder in das Buch steckst.

Vorwort von Peer Heinlein

Als ich 1992 als Jugendlicher eine Computermailbox zu meinem Hobby machte, kam mir nie in den Sinn, dass dies auch fast 30 Jahre später noch mein täglicher Lebensinhalt sein würde.

Was anfangs noch ein MS-DOS-System war, wurde schon wenig später auf das damals noch revolutionäre und brandneue Linux-System umgerüstet. Den Um- und Irrweg über ein Windows-System habe ich darum nie gehen müssen – weder auf Servern noch auf meinen Privatcomputern –, und vermutlich liegt es daran, dass ich bis heute ein halbwegs entspanntes Verhältnis zu meinen Kisten habe. Sie machen schließlich das, was sie machen sollen. Meistens.

Die Vernetzung von Menschen und der freie Fluss der Kommunikation sind seitdem mein Lebensinhalt geworden. Seit rund 30 Jahren bin ich darum als Trainer dabei, anderen Administratoren die Linux-Systemadministration zu vermitteln: technische Fakten, vor allem aber auch »Softskills«, also Kompetenzen rund um das technische Verständnis der Abläufe, die Fähigkeit, sich selbst neue Themen zu erarbeiten, sicher und zielgerichtet Fehler einzukreisen und Fehlverhalten zu debuggen. Die Arbeit mit Menschen ist es, die Spaß macht. Computer selbst sind kein Selbstzweck, sie sind nur Mittel zum Zweck: Arbeitstiere. Aber praktische Arbeitstiere, wenn man sie effizient und sicher einsetzt.

In diesem Werk habe ich vor allem die Verantwortung für die Mailserver-Kapitel übernommen, schließlich habe ich bereits einige Fachbücher rund um Postfix und Dovecot veröffentlicht. In diesem Administrationshandbuch haben wir die Gelegenheit genutzt, statt eines umfassenden vollständigen Nachschlagewerks eine klare, nachvollziehbare Anleitung zum Aufbau eines eigenen Mailsystems auszuarbeiten, ohne allzu viel Grundlagenwissen vorauszusetzen oder den Anspruch zu haben, den perfekten Postmaster auszubilden.

Dazu gehört natürlich auch im Bereich Anti-Spam/Anti-Virus ein Kapitel zu »rspamd«, denn das hat sich in den letzten Jahren vom Geheimtipp zum Standard in diesem Bereich entwickelt. Wir haben ihm viel zu verdanken – es hält nicht nur die Postfächer sauber von nervigem Spam, es beschützt auch uns und unsere Daten: Denn es übernimmt auch den Kampf gegen Viren und per Mail verschickte erpresserische Ransomware und hat hier sicherlich schon viel Ärger und (bei verschlüsselten Dateien?) auch Leid verhindert.

Eine ebensolche Allzweckwaffe ist die bewährte Monitoring-Lösung »Checkmk«: Damit überwachen wir nicht nur plumpe Verfügbarkeiten von Diensten, sondern erfassen auch fortlaufend viele qualitative Messwerte vieler, vieler kleiner Details unserer Server. Das ist wichtig für die Analyse komplexer und verwirrender Fehlerbilder, notwendig für eine spätere Informationsgewinnung, wie es zu Beeinträchtigungen kommen konnte. Über Wochen hinweg erfasste Daten, automatisch grafisch aufbereitet. Eine wirklich mächtige Software, mit der zu arbeiten einfach Spaß macht.

Aber Spaß kann auch schnell vorbei sein. Getreu dem Motto »Zuerst hatten wir kein Glück, und dann kam auch noch Pech hinzu« habe ich auch das Kapitel »Backup und Recovery« zu verantworten, denn mit »Relax & Recover (ReaR)« kann die von vielen Administratoren so vernachlässigte Disaster Recovery bequem Einzug finden. Also: Vergessen Sie vor lauter Euphorie über Aufbau und Überarbeitung Ihrer Linux-Systeme nicht, rechtzeitig an den Plan B zu denken, falls es mal schiefgeht! Vielen Dank an Schlomo Schapiro für ReaR – und auch für die Fachkontrolle meines Backup-Kapitels.

Danksagung

Am »rspamd«-Kapitel haben meine Kollegen Carsten Rosenberg und Manu Zurmühl mit viel Aufwand mitgearbeitet und Anleitungen sowie Schulungsmaterial beigeleitet. Am »Checkmk«-Teil hat mein Kollege Robert Sander geduldig mit vielen Praxistipps und seiner ganzen Erfahrung aus- und nachgeholfen. Mit unseren weiteren Consulting-Kollegen Mirko Ludeke, Leah Herbach und Torsten Lange bilden alle zusammen ein starkes Team, auf das man sich stets verlassen kann und das, so kann man schon sagen, unseren Kunden in jedem Notfall schnell und kompetent hilft und eigentlich stets als Sieger aus einem Troubleshooting hervorgeht. Vielen Dank für Eure Geduld, Unterstützung und Hilfe - Ihr seid ein ganz wesentlicher Teil von »Heinlein« und bietet Rückendeckung und Verlässlichkeit für so viele Linux-Admins da draußen. Und bei dieser Gelegenheit einen ganz persönlichen Dank an Robert dafür, dass Du nun schon so lange für mich der Fels in der Brandung bist.

Vielen Dank an die aktuellen und früheren Autoren Charly, Daniel, Dirk und vor allem auch an Stefan Kania. Auch mit ihm arbeite ich seit rund 20 Jahren zusammen und habe ihn nicht nur als fachlich jederzeit hochkompetenten Spezialisten, sondern auch privat sehr zu schätzen gelernt. Vielen Dank für diesen doch schon immens langen und tollen Weg, den wir zusammen gehen.

Der Dank an mein Team hier bei Heinlein Support kann gar nicht groß genug sein, denn Ihr haltet mir in so vielen Bereichen den Rücken frei, damit ausreichend Zeit bleibt, auch Projekte wie dieses Buch voranzutreiben. Vielen Dank für alles, was wir gemeinsam im Team leisten und was auch jeder Einzelne bewegt, vorantreibt, korrigiert und gestaltet.

Zu guter Letzt danke ich meiner Frau Ivonne und meinen Kindern Antonia und Carolin für alles und uns als Familie dafür, dass alles so ist, wie es ist!

Vorwort von Daniel van Soest

Wie die Jungfrau Maria zum Kind, so bin ich zu diesem Buch gekommen – oder eher dazu, ein Koautor dieses Buches zu werden. Nun halten Sie bereits die siebte Auflage in den Händen; davon hätte ich vor fast zehn Jahren nicht einmal zu träumen gewagt.

Der Praxisbezug ist mir sehr wichtig, ebenso wie das Aufbauen von Hintergrundwissen. Während meiner nun mehr als 20-jährigen Berufserfahrung im Kommunalen Rechenzentrum Niederrhein (KRZN) durfte ich viele Hürden überwinden, aber noch mehr Erfolge feiern. Ich habe in diesem Buch stets versucht, nicht nur die Technik zu erläutern, sondern auch einen Großteil meiner Erfahrung mit einfließen zu lassen. Für dieses Buch war einer meiner Leitsätze: »Man kann nur die Technik beherrschen, die man versteht.«

Ich hoffe, diesem Motto gerecht geworden zu sein und mit diesem Buch nicht nur eine Anleitung geschaffen zu haben, sondern Sie darin unterstützen zu können, die Technik zu verstehen, selbst kreative Ideen zu entwickeln und nicht nur stumpf nach Plan zu arbeiten.

Abschließend bleibt mir nur noch eins: Ihnen viel Spaß mit diesem Buch zu wünschen.

Danksagung

Vorab möchte ich mich bei meinen Koautoren Dirk, Stefan, Peer und Axel bedanken. Die Zusammenarbeit war sowohl kreativ als auch produktiv, auch wenn wir die eine oder andere Hürde meistern mussten: Das Ergebnis kann sich sehen lassen. Ebenso möchte ich mich bei Christoph Meister bedanken – ohne Dein Lektorat, die Geduld und die guten Lösungsansätze wären wir jetzt nicht da, wo wir sind. Nicht vergessen werden darf auch Norbert Englert: Vielen Dank für die schönen Bilder und noch viel mehr für die \LaTeX - und Satz-Unterstützung. Natürlich darf die Korrektorin nicht vergessen werden – vielen Dank, Frau Daenecke. Ebenso geht mein Dank an meine Band (4Dirty5): Danke, dass Ihr mir die Möglichkeit gebt, den Ausgleich zu bekommen, den ich zum Alltag brauche, und dass ihr meine gelegentliche Abwesenheit verkraftet habt.

Zum Abschluss möchte ich mich bei den wichtigsten Personen in meinem Leben bedanken, ohne viele Worte zu bemühen: Ich bin dankbar, Euch meine Familie nennen zu dürfen. Danke, Nicole, danke Tom, danke Linda!

Vorwort von Axel Miesen

Zur fünften Auflage des Linux-Server-Handbuchs im Jahr 2019 durfte ich zum ersten Mal zwei Kapitel zu den Themen Ansible und Docker beisteuern; nun haben wir bereits die siebte Auflage erreicht, und ich freue mich immer noch sehr, bei diesem Projekt und diesem Team dabei zu sein.

Auch dieses Mal gibt es in meinen Kapiteln im Vergleich zur vorherigen Auflage wieder zahlreiche Änderungen. Beim Thema Ansible habe ich mich entschieden, viele Grundlagen noch

ausführlicher darzustellen und dafür (hauptsächlich aus Platzgründen) auf die Unterthemen »Rollen« und »Ansible Vault« zu verzichten bzw. nur einen kleinen Ausblick darauf zu geben.

Das Kapitel über Docker trägt nun den neuen Titel »Containervirtualisierung mit Docker und Podman«, womit eine weitere Neuerung bereits deutlich wird: Die Docker-Alternative Podman, die sich immer größerer Beliebtheit erfreut, hat nun auch in diesem Buch einen angemessenen Raum bekommen. Zudem habe ich den Unterabschnitt über private Registries sehr vereinfacht; ich setze dort nun auf die freie Software Harbor, die im Vergleich zur klassischen Docker Registry kaum Wünsche offen lässt und zudem sehr schnell an den Start zu bringen ist.

Ich hoffe, dass Ihnen mit meinen Kapiteln die ersten Schritte im Konfigurationsmanagement und in der schönen (nicht mehr ganz so neuen) Containerwelt leichter fallen werden, und wünsche Ihnen viel Spaß und Erfolg!

Danksagung

Mein großer Dank gilt zunächst meinen Autorenkollegen, die dieses Buch in vielen Jahren zu einem großen Erfolg geführt haben und mir auch immer mit nützlichen Tipps weitergeholfen haben. Auch dem Dank ans Rheinwerk-Team, namentlich Christoph Meister, Friederike Daenecke und Norbert Englert, möchte ich mich unbedingt anschließen.

Nicht zuletzt danke ich den beiden wichtigsten Personen in meinem Leben: meiner Lebensgefährtin Ana und meiner Tochter Lena. Danke, dass ihr immer für mich da seid, und dass ihr stets Verständnis dafür hattet, wenn ich mitunter auch am Wochenende mal an diesem Buch gearbeitet habe.

Über dieses Buch

An dieser Stelle möchten wir Ihnen erklären, was wir uns bei der Verwendung der verschiedenen Formatierungsmöglichkeiten gedacht haben. Hier finden Sie auch die Beschreibung zu den im Buch verwendeten Icons und die Begründung, warum wir uns gerade für diejenigen Distributionen entschieden haben, die im Buch verwendet werden.

Formales

Damit Sie den größtmöglichen Nutzen aus diesem Buch ziehen können, verwenden wir einige formale Konventionen, die im Folgenden erläutert werden.

Kommandozeile

Gleich zu Beginn ein Hinweis an den mausverwöhnten Windows-Nutzer: Wir werden im Rahmen dieses Buches hauptsächlich Gebrauch von der Kommandozeile machen, da sich viele Aufgaben unter Linux einfacher und ökonomischer durch einige Tastaturkommandos erledigen lassen. Nur in einem Kapitel stehen die grafischen Werkzeuge mehr im Vordergrund, und zwar im Samba-4-Kapitel. Auch als Linux-Admin werden Sie dort die grafischen Werkzeuge benötigen, denn nicht alle Aufgaben können über die Kommandozeile realisiert werden: Wenn Sie eine Active Directory-Domäne verwalten wollen, kommen Sie an den grafischen Werkzeugen nicht vorbei.

Das soll allerdings nicht heißen, dass wir gänzlich auf den Komfort einer grafischen Umgebung verzichten, denn wie bei vielen Dingen im Leben gilt auch hier: Die Mischung macht's.

Für viele Bereiche gibt es heute grafische Werkzeuge, gerade webbasierte, die Ihnen als Administrator das Leben leichter machen können. Auch wir nutzen diese Werkzeuge und werden an den entsprechenden Stellen auf sie eingehen.

Befehle eingeben

Für Kommandozeilenbefehle soll folgende Schreibweise verwendet werden: Im fließenden Text werden Konsolenbefehle durch Nicht-Proportionalschrift gekennzeichnet. Viele Beispiele zu den Kommandos werden aber auch in Listings dargestellt und in Nicht-Proportionalschrift wiedergegeben. In den Listings können Sie von der Befehlszeile bis zum Ergebnis alles nachvollziehen:

```
stefan@adminbuch~$ ps
PID TTY          TIME CMD
 4008 pts/2      00:00:00 bash
 4025 pts/2      00:00:00 ps
```

Listing 1 Beispiel für ein Listing

Privilegierte Rechte

Für die Administration von Linux-Systemen werden Sie immer root-Rechte benötigen, um die entsprechenden Konfigurationsdateien bearbeiten oder um Dienste starten oder stoppen zu können.

Ubuntu vertritt im Unterschied zu anderen Linux-Distributionen eine eigene Philosophie: Der Standardbenutzer der ersten Installation kann jeden Administratorbefehl durch Voranstellen des Befehls `sudo` ausführen. Anschließend muss dann das Passwort des Standardbenutzers eingegeben werden:

```
stefan@adminbuch~$ sudo systemctl restart systemd-networkd
[sudo] password for <user>: <Hier eigenes Passwort eingeben>
```

Listing 2 Arbeiten als root

Sind mehrere Befehle als Administrator einzugeben, so kann das Voranstellen von `sudo` auch lästig werden. In diesem Fall verschaffen Sie sich mit dem folgenden Befehl vorübergehend eine root-Shell:

```
stefan@adminbuch~$ sudo -s
[sudo] password for <user>: <Hier eigenes Passwort eingeben>
root@adminbuch~#
```

Listing 3 Eine root-Shell öffnen unter Ubuntu

Eingabe langer Befehle

Und noch eine weitere wichtige, eher technische Konvention: Einige der vorgestellten Kommandozeilenbefehle oder Ausgaben von Ergebnissen erstrecken sich über mehrere Buchzeilen. Im Buch kennzeichnet am Ende der entsprechenden Zeilen ein »\«, dass der Befehl oder die Ausgabe in der nächsten Zeile weitergeht. Geben Sie das Kommando auf der Konsole ohne den Backslash und ohne Zeilenumbruch ein.

Screenshots

Wie heißt es doch so schön: Ein Bild sagt mehr als tausend Worte. Wann immer es sinnvoll erscheint, soll daher ein Screenshot zur Erhellung des Sachverhalts beitragen.

Internetverweise

Da wir in diesem Buch sehr viele verschiedene Dienste ansprechen, ist es nicht möglich, alle Funktionen und Fähigkeiten eines Dienstes bis ins kleinste Detail zu beschreiben. Aus diesem Grund haben wir an geeigneten Stellen auf Internetadressen verwiesen. Verweise auf Internetadressen werden besonders ausgezeichnet, zum Beispiel so: www.debian.org

Icons

Sie werden in den einzelnen Kapiteln am Rand häufig Icons finden, die Sie auf bestimmte Zusammenhänge oder Besonderheiten hinweisen sollen. Die Icons haben die folgenden Bedeutungen:

Hier wird es immer sehr wichtig

Wann immer Sie das nebenstehende Symbol sehen, ist Vorsicht angeraten: Hier weisen wir auf besonders kritische Einstellungen hin oder auf Fehler, die dazu führen können, dass das System nicht mehr stabil läuft. Damit sich die Warnungen deutlich vom restlichen Text abheben, haben wir diese Textbereiche zusätzlich mit einem grauen Kasten hinterlegt.



Beispiele – etwa für Konfigurationsdateien – haben wir mit diesem Symbol gekennzeichnet. Wir haben an vielen Stellen Beispiele eingefügt, die es Ihnen leichter machen, eine entsprechende Aufgabe umzusetzen.



Alle Textstellen, die wir mit diesem Icon versehen haben, sollten Sie unbedingt lesen! Hier handelt es sich um wichtige Hinweise zu den unterschiedlichen Distributionen, die wir verwenden, oder um wichtige Eigenschaften oder Konfigurationsmöglichkeiten eines Dienstes.



Es gibt keine fehlerfreie Software! Große und kleine Fehler, die bei den einzelnen Diensten bekannt sind, werden durch diesen kleinen »Bug« gekennzeichnet. Die nachweislich erste Erwähnung des Wortes »Bug« stammt übrigens von Grace Hopper, einer Computerpionierin aus den USA: http://de.wikipedia.org/wiki/Grace_Hopper



Bei diesem Symbol finden Sie nützliche Tipps und Tricks zu bestimmten Aufgaben.



Linux-Distributionen

Als damals der Gedanke zur ersten Auflage für dieses Buch aufkam, mussten wir uns erst einmal einig werden, welche Distributionen wir denn für das Buch verwenden wollten. Aufgrund der folgenden Kriterien haben wir dann unsere Entscheidung getroffen:

- ▶ Wir wollten auf jeden Fall mindestens eine Distribution, die *rpm*-Pakete, und eine, die *deb*-Pakete für die Softwareverwaltung nutzt.
- ▶ Da es in diesem Buch um Serverdienste geht, musste die Distribution nicht unbedingt die aktuellsten Pakete haben, wie man es gerne auf einem Desktop hat, sondern uns kam es in erster Linie auf die Stabilität an. Dennoch haben wir bei manchen Diensten durchaus auf eine bestimmte minimale Versionsnummer geachtet.
- ▶ Die Distributionen sollten sehr verbreitet sein und oft in Firmen zum Einsatz kommen.
- ▶ Der Supportzeitraum sollte mindestens vier bis fünf Jahre betragen, also ungefähr die Laufzeit, die IT-Systeme in Unternehmen haben.

Aufgrund dieser Kriterien haben wir uns im Laufe der Zeit immer wieder Gedanken gemacht, welche Distribution wir einsetzen, so auch dieses Mal. Dabei ist die Auswahl auf die folgenden Distributionen gefallen:

► **Debian Bullseye**

Debian ist seit Jahren für stabile Versionen und hohe Zuverlässigkeit bekannt. Auch ist die Bereitstellung der Sicherheitsupdates für einen langen Zeitraum gesichert.

► **openSUSE Leap**

Viele Leser haben uns gefragt, warum wir nicht mehr mit openSUSE arbeiten, und wir sehen auch, dass die openSUSE-Distributionen, auch in Unternehmen, immer öfter eingesetzt werden. Gerade wenn es um Desktop-Systeme in Domänen geht, wird openSUSE oft verwendet. Deshalb haben wir uns auch im Samba-4-Kapitel dafür entschieden, openSUSE Leap als grafischen Client einzusetzen.

► **Ubuntu-Server 22.04 LTS**

Der Ubuntu-Server basiert auf Debian und stellt mit der *LTS*-(*Long Term Support*-)Version eine gute Alternative zum Debian-Server dar. Der Ubuntu-Server setzt dabei auf neuere Pakete und Kernel als Debian, da bei Ubuntu die Releasezyklen kürzer sind.

► **CentOS Stream**

CentOS wird auch in dieser Auflage genutzt, und zwar die Version *Stream*. Bei CentOS Stream handelt es sich um ein *rolling release*, das bedeutet, dass es keine neuen Versionen mehr geben wird, sondern die bestehende Version immer aktualisiert wird. Bei dieser neuen Edition hat es die verschiedensten Änderungen gegeben, auch hinsichtlich der unterstützten Software. So ist zum Beispiel OpenLDAP nicht mehr Bestandteil der Distribution. Wir haben lange überlegt, ob und welche Version von CentOS wir ins Buch aufnehmen, und dann für CentOS Stream entschieden. Wir hoffen, dass auch in Zukunft alles das, was wir im Buch beschrieben haben, weiterhin möglich sein wird.

Wenn Sie sich jetzt fragen: »Aber meine Lieblingsdistribution erfüllt die Punkte auch, warum ist die nicht dabei?«, können wir an dieser Stelle nur sagen, dass wir natürlich alle Dienste unter allen von uns verwendeten Distributionen konfiguriert und ausgetestet haben. Allein für das Testen mit vier verschiedenen Distributionen benötigt man schon eine geraume Zeit. Deshalb haben wir uns für diese vier Distributionen entschieden.

Jetzt bleibt uns nur noch, Ihnen viel Spaß mit dem Buch zu wünschen und zu hoffen, dass Ihnen unser Buch bei Ihrer täglichen Arbeit eine Hilfe sein wird.

Kapitel 1

Der Administrator

In diesem Kapitel geht es um den Beruf des Administrators, um notwendige Fähigkeiten und Fertigkeiten zu seiner Ausübung, eine Einordnung der eigenen Tätigkeit, Verhaltensempfehlungen und um einen Ehrenkodex.

Vielen Administratoren fällt es schwer, ihre Arbeit einzuordnen und zu planen. Dieses Kapitel soll Ihnen Unterstützung bieten, diese Einordnung vorzunehmen. Außerdem bekommen Sie einige hilfreiche Hinweise zur Kommunikation mit Kollegen und Vorgesetzten. Eine bewährte Planungsmethode zur Gestaltung des Arbeitstages rundet neben dem Ehrenkodex dieses Kapitel ab.

1.1 Der Beruf des Systemadministrators

Der Systemadministrator wird selten wahrgenommen. Meist wird erst im Fehlerfall bemerkt, dass es jemanden gibt, der sich um die Dienste kümmert, die jeder täglich nutzt. Es wäre jedoch falsch, zu sagen, dass nur der Systemadministrator ein guter Systemadministrator ist, der nicht wahrgenommen wird.

Aber genau in diesem Spannungsfeld liegt eine der sehr großen nicht technischen Herausforderungen dieses Berufszweigs, und das zeigt sich direkt auch schon daran, dass Kommunikation ein wichtiger Punkt in der Liste der Fähigkeiten ist, die einen Systemadministrator auszeichnen. Auf die angeforderten Fähigkeiten kommen wir im weiteren Verlauf des Kapitels noch zu sprechen. Um die gleiche Sprache zu verwenden, werden zunächst einige gebräuchliche Begriffe aus dem Arbeitsumfeld des Systemadministrators erklärt.

1.1.1 Berufsbezeichnung und Aufgaben

Da »Systemadministrator« kein geschützter Begriff und auch kein Lehrberuf ist, herrscht immer Verwirrung darüber, welche Ausbildungsgänge oder Tätigkeitsfelder dem Berufsbild zugeordnet werden. Häufig finden sich in dieser Berufsgruppe Quereinsteiger aus informatikfernen Berufen. Allgemeiner Konsens ist allerdings, dass den Administratoren folgende Tätigkeiten zugeordnet werden:

► **Installation**

von Systemen, Hardware und Software. Hier geht es vor allem um Arbeitsplatzrechner und Server, aber auch Speichersysteme (Backup, NAS und SAN) sowie Netzwerkkomponenten können dazugehören.

► **Konfiguration**

von Systemen. Dies bedeutet sowohl das Einstellen der Systeme auf Benutzerbedürfnisse, als auch die Optimieren von Konfigurationen in Bezug auf die Leistung.

► **Betrieb**

von Systemen, das Sicherstellen der Funktionsfähigkeit und die nachhaltige Problemlösung im Fehlerfall

► **Anlegen**

von Benutzern nach gesetzlichen Vorgaben und den Richtlinien des Unternehmens

► **Vergabe und Rücknahme**

von Benutzerrechten

► **Beratung**

bei der Hard- und Softwareanschaffung (nach den Erfordernissen und dem Budget)

► **Unterstützung**

bei Projekten der Informationstechnologie

Je nach Größe der Firma, in der ein Systemadministrator tätig ist, kann die Arbeit sehr spezialisiert sein und nur Teile der oben angeführten Aufgaben berühren. In kleineren Unternehmen gibt es Überschneidungen mit dem, was zum Aufgabengebiet eines Netzwerkadministrators, eines Datenbankadministrators oder eines Anwendungsbetreuers zählt. Im Großrechner-Umfeld spricht man häufig vom *Systemprogrammierer* und vom *Operator*, in der Open-Source-Welt finden sich beide Berufsbilder häufig in den Aufgaben der Systemadministratoren wieder.

1.1.2 Job-Definitionen

In diesem Abschnitt fassen wir die allgemein anerkannten Berufsbezeichnungen nach der *System Administrators Guild* der *Usenix Association* (SAGE) zusammen, die ebenfalls von der *League of Professional System Administrators* (LOPSA) anerkannt werden. Vergleichen Sie: https://www.usenix.org/system/files/lisa/books/usenix_22_jobs3rd_core.pdf

Die Definitionen der folgenden Richtlinien sind nicht in Stein gemeißelte Gesetze, aber sie helfen, eine allgemeine Kategorisierung des Wissensstandes von Systemadministratoren vorzunehmen. Diese Kategorisierung ist nicht auf Linux- oder UNIX-Administratoren beschränkt, beschreibt diese aber auch.



Der Stoff ist zwar ein bisschen trocken, aber die Lektüre lohnt sich, um den eigenen Standort bestimmen zu können.

Unternehmensgröße

Bei der Größe der Unternehmungen, in der die Systemadministratoren zu finden sind, werden grob drei verschiedene Ausbaustufen unterschieden:

1. Man spricht von einem **kleinen einheitlichen Betrieb**, wenn weniger als 50 Computer mit dem gleichen Betriebssystem im Einsatz sind und weniger als 20 Nutzer an ihnen arbeiten. Die Computer der Administratoren werden hierbei nicht mitgerechnet.
2. Ein **komplexer Betrieb** hat bis zu 100 Computer, und die bis zu 100 Benutzer arbeiten mit mehr als einem Betriebssystem.
3. Den **großen komplexen Betrieb** kennzeichnen mehr als 100 Computer und mehr als 100 Benutzer mit mehr als einem Betriebssystem.

Das ist nur eine grobe Einteilung, die aber eine ungefähre Richtlinie festlegt. Eine Kombination dieser einzelnen Betriebsarten gibt es immer, und die Grenzen sind fließend.

Novice System Administrator

► Erforderliche Fähigkeiten

- hat ein hohes Maß an sozialer Kompetenz und an Kommunikationsfähigkeiten; die Fähigkeit, einfache Sachverhalte schriftlich oder mündlich zu erläutern; gute Fähigkeiten am Telefon
- ist vertraut mit einem Betriebssystem und dessen Befehlen und Werkzeugen auf Benutzerebene; ist in der Lage, Dateien zu editieren, Kommandos auszuführen, Homeverzeichnisse der Nutzer zu finden, durch das Dateisystem zu navigieren und Ein-/Ausgabeumlenkung einzusetzen
- kann Anweisungen gut folgen

► Erforderliche Ausbildung

- zwei Jahre an der Hochschule, eine äquivalente Ausbildung oder Berufserfahrung nach der Schule

► Wünschenswerte Ausbildung und Fähigkeiten

- ein Abschluss oder eine Zertifizierung in Informatik (oder einem verwandten Bereich)
- vorhergehende Erfahrungen im Kundendienst, Rechnerbetrieb, in der Systemadministration oder in einem verwandten Bereich
- Motivation, sich beruflich weiterzuentwickeln

► Angemessene Verantwortlichkeiten

- führt Routineaufgaben unter direkter Aufsicht eines erfahreneren Administrators aus
- fungiert als Direktkontakt zu den Nutzern, nimmt Fehlermeldungen an und weist sie den entsprechenden Systemadministratoren zu

Junior System Administrator

► Erforderliche Fähigkeiten

- hat ein hohes Maß an sozialer Kompetenz und an Kommunikationsfähigkeiten; ist in der Lage, Benutzern Anwendungen und Betriebssystem-Grundlagen beizubringen sowie eine Basis-Dokumentation zu schreiben
- Fähigkeit, die meisten Betriebssystem-Kommandos und -Werkzeuge einzusetzen
- ist vertraut mit den meisten Basis-Werkzeugen der Systemadministration und den Basis-Prozessen; ist beispielsweise in der Lage, eine Maschine zu starten und herunterzufahren, Benutzerkonten hinzuzufügen und zu entfernen, Backupprogramme zu nutzen, Filesystem- und Datenträgertests durchzuführen und Systemdatenbanken zu pflegen (Nutzer, Gruppen, Hosts, Alias)
- besitzt ein Grundverständnis des Betriebssystems; hat beispielsweise das Prinzip der Job-Steuerung, Soft- und Hardlinks oder Verknüpfungen verstanden, kann zwischen Betriebssystem-Kern und Nutzerumgebung unterscheiden

► Erforderlicher Hintergrund

- ein bis drei Jahre Erfahrung in der Systemadministration

► Wünschenswerter Hintergrund und Fähigkeiten

- Abschluss in Informatik oder einem verwandten Feld
- ist mit den Konzepten vernetzter und verteilter Computerumgebungen vertraut; kann beispielsweise das route-Kommando benutzen oder das Routing und die Dienste für den Fernzugriff verwalten, kann Workstations zum Netzwerk hinzufügen und entfernte Dateisysteme einbinden
- ist in der Lage, Skripte in einer oder mehreren Verwaltungssprachen wie Tk, Perl, Python, VBScript oder als Shell-Skript zu schreiben
- hat Programmiererfahrung in der passenden Programmiersprache

► Angemessene Verantwortlichkeiten

- alleinige Verwaltung einer kleinen einheitlichen Niederlassung oder Unterstützung in der Administration einer größeren Systemumgebung
- arbeitet unter der Aufsicht eines Systemadministrators oder eines Managers

Intermediate/Advanced System Administrator

► Erforderliche Fähigkeiten

- hat ein hohes Maß an sozialer Kompetenz und an Kommunikationsfähigkeiten; ist in der Lage, Begründungen für Kaufanträge zu schreiben, Nutzer in komplexen Inhalten zu schulen, Präsentationen vor einem internen Publikum zu halten, sich mit dem oberen Management auseinanderzusetzen

- unabhängiges Lösen von Problemen, selbstständiges Arbeiten
 - ist vertraut mit den meisten Aspekten der Betriebssystem-Administration; beispielsweise mit dem Konfigurieren von Mail-Systemen, mit der Betriebssystem-Installation und -Konfiguration, mit der Einrichtung von Druckern, mit den Grundlagen der Security und der Installation der Software von Drittanbietern
 - hat ein umfassendes Verständnis von UNIX-basierten Betriebssystemen; versteht Paging und Swapping, die Kommunikation zwischen Prozessen, die Geräte und was Gerätetreiber tun, kennt Dateisystemkonzepte (*inode, clustering, logical partitions*)
 - ist mit den grundlegenden Konzepten von vernetzten und verteilten Rechnerumgebungen vertraut; kann NFS-, NIS- und NT-Domänen konfigurieren, kann `nslookup` oder `dig` benutzen (DNS); versteht grundlegende Routing-Konzepte
 - ist in der Lage, Skripte in einer oder mehreren Verwaltungssprachen wie Tk, Perl, Python, VBScript oder als Shell-Skript zu schreiben
 - ist in der Lage, minimales Debugging von und kleine Veränderungen an C-Programmen durchzuführen
- ▶ **Erforderlicher Hintergrund**
- drei bis fünf Jahre Erfahrung in der Systemadministration
- ▶ **Wünschenswerter Hintergrund und Fähigkeiten**
- Abschluss in Informatik oder einem verwandten Feld
 - bedeutende Kenntnisse in der passenden Programmiersprache
- ▶ **Angemessene Verantwortlichkeiten**
- bekommt grundlegende Anweisungen für neue Verantwortlichkeiten von einem Vorgesetzten
 - administriert einen komplexen Betrieb allein oder unterstützt die Administration eines größeren Betriebs
 - initiiert und übernimmt einige neue Verantwortlichkeiten und hilft bei der Zukunftsgestaltung des Betriebs oder des Netzwerks
 - betreut *Novice System Administrators* oder *Operators*
 - beurteilt und/oder befürwortet Neuanschaffungen; hat starken Einfluss auf Kaufentscheidungen

Senior System Administrator

- ▶ **Erforderliche Fähigkeiten**
- hat umfassende Kenntnisse der Konzepte von vernetzten und verteilten Rechnerumgebungen; versteht Routing, Client/Server-Programmierung; ist fähig zum Design von beständigen, netzwerkweiten Dateisystemen

- ist in der Lage, Probleme schnell zu lösen und Prozesse zu automatisieren
 - hat ein hohes Maß an sozialer Kompetenz und an Kommunikationsfähigkeiten; ist in der Lage, Anträge oder Berichte zu schreiben, fungiert als Kontaktperson für Vertriebsbeauftragte, hält Präsentationen für Kunden, Auftraggeber oder professionelle Partner, arbeitet eng mit dem oberen Management zusammen
 - besitzt umfassende Kenntnisse in einem Betriebssystem, versteht Paging und Swapping, die Kommunikation zwischen Prozessen, die Geräte und was Gerätetreiber tun, kann Performance-Analysen nutzen, um Systeme einzustellen und kennt Dateisystemkonzepte (*inode, clustering, logical partitions*)
 - ist in der Lage, Skripte in einer Verwaltungssprache wie Tk, Perl, Python, VBScript oder als Shell-Skript zu schreiben, C-Programme von einer Plattform auf eine andere zu portieren und kleine C- oder C#-Programme zu schreiben
- ▶ **Erforderlicher Hintergrund**
- mehr als fünf Jahre Erfahrung in der Systemadministration
- ▶ **Wünschenswerter Hintergrund und Fähigkeiten**
- Abschluss in Informatik oder einem verwandten Feld
 - weitreichende Kenntnisse in der passenden Programmiersprache
 - Publikationen im Bereich der Systemadministration
- ▶ **Angemessene Verantwortlichkeiten**
- gestaltet/implementiert komplexe lokale oder weitreichende Netzwerke von Maschinen
 - leitet einen großen komplexen Betrieb oder ein entsprechendes Netzwerk
 - arbeitet gewöhnlich unter der Leitung des Senior Managements
 - etabliert oder empfiehlt Richtlinien zur System- und Dienstenutzung
 - bietet technische Führung und/oder beaufsichtigt Systemadministratoren, Systemprogrammierer oder eine andere entsprechende Führungsebene
 - hat Kaufbefugnis und Verantwortung für Einkäufe

1.1.3 Definitionen der Management-Level

In den »Core Job Descriptions«¹ findet sich auch die Beschreibung der Management-Level. Auch wenn das nicht direkt mit Systemadministratoren zu tun hat, ist es doch hilfreich, die einzelnen Aufgaben der Leitungsebenen zu verstehen. Wir erwähnen die Verantwortlichkeiten der einzelnen Level aus Gründen der Vollständigkeit.

1 https://www.usenix.org/system/files/lisa/books/usenix_22_jobs3rd_core.pdf

Technical Lead

- ▶ unterstützt das Team durch seine Mitarbeit
- ▶ automatisiert sich wiederholende Tätigkeiten, wo immer es geht, um es dem Team zu ermöglichen, mit dem Wachstum der Organisation Schritt zu halten
- ▶ steuert und arbeitet in der Systemadministration
- ▶ assistiert dem System Administration Manager beim Setzen der Ziele und im Training, beim Definieren von Technologieprioritäten und bei der Entwicklung einer Langzeitstrategie, um die Systemadministration zu dimensionieren und zu leiten
- ▶ betreut einen oder mehrere Mitarbeiter, agiert als Mentor und gibt ihnen technische Führung
- ▶ kommuniziert und handelt als Bindeglied zwischen Endbenutzern und Kollegen
- ▶ fungiert als Bindeglied zwischen Teammitgliedern und dem System Administration Manager
- ▶ kommuniziert die Entwicklung der Prioritäten und des Budgets in Richtung des Teams und des Managements

System Administration Manager

- ▶ legt die Ziele des Teams fest, definiert Technologieprioritäten und entwickelt langfristige Strategien zur Führung und Entwicklung der Systemadministration in der Organisation
- ▶ betreut einen oder mehrere Mitarbeiter, agiert als Mentor und gibt ihnen technische Führung
- ▶ liefert Karriereunterstützung und Performance-Feedback an Teammitglieder
- ▶ kommuniziert und handelt als Bindeglied zwischen Endbenutzern und Kollegen
- ▶ fungiert als Bindeglied zwischen Teammitgliedern und dem IT Director
- ▶ kommuniziert die Entwicklung der Prioritäten und des Budgets in Richtung des Teams und des Managements

IT Director

- ▶ plant und gibt die taktische Richtung vor, setzt Managementziele, definiert Prioritäten und entwickelt langfristige Strategien zur Führung und Entwicklung der Systemadministration in der Organisation
- ▶ entwickelt, integriert und verwaltet eine rund um die Uhr verfügbare IT-Umgebung, stellt Erweiterbarkeit, Integrität, Performance, Wirtschaftlichkeit und Zuverlässigkeit sicher
- ▶ leitet den Lieferanten-Auswahlprozess, verhandelt Verträge und managt bestehende Beziehungen und Lieferungen

- ▶ beaufsichtigt einen oder mehrere System Administration Manager und unterstützt sie mit taktischen Orientierungshilfen und Mentoring
- ▶ bietet Karriereunterstützung und beurteilt die Leistung von direkten Untergebenen
- ▶ kommuniziert mit Partnern und dem Management quer durch die Organisation, um sicherzustellen, dass infrastrukturbezogene Prioritäten mit Organisationszielen und -bedürfnissen gekoppelt werden
- ▶ handelt als Kontaktperson zwischen IT-Managern und CIO oder dem Senior Management
- ▶ kommuniziert die Entwicklung von Prioritäten und Budget in Richtung Senior Management und zu den direkten Untergebenen

Chief Information Officer

- ▶ plant und bestimmt die Richtung, setzt Managementziele, definiert Prioritäten und entwickelt langfristige Strategien zur Leitung und Dimensionierung von sicheren und zuverlässigen IT-Arbeiten für die Organisation
- ▶ leitet direkt einen oder mehrere Manager und bietet strategische Führung und Vision
- ▶ bietet Karriereunterstützung und beurteilt die Leistung von direkten Untergebenen
- ▶ kommuniziert mit Partnern quer durch die Organisation, dem Firmenvorstand und Senior Management, um sicherzustellen, dass IT-bezogene Absichten mit Organisationszielen gekoppelt werden und einen Wettbewerbsvorteil für die Organisation bringen
- ▶ fungiert als Ansprechpartner für die Bedürfnisse der IT zwischen Firmenvorstand und Senior Management und Organisationseinheiten
- ▶ kommuniziert die Entwicklung von Prioritäten und Budget sowohl in Richtung Firmenvorstand und zum Senior Management als auch zu den direkten Untergebenen

1.2 Nützliche Fähigkeiten und Fertigkeiten

Abseits von den Fachkenntnissen, die elementar sind, um den Beruf eines Systemadministrators ausüben zu können, kommen jetzt Fähigkeiten zur Sprache, die man vielleicht nicht direkt in Zusammenhang mit den Anforderungen bringt, die einen Systemadministrator auszeichnen, die aber dennoch wichtig für die Ausübung dieses Berufs sind.

1.2.1 Soziale Fähigkeiten

Systemadministration hat nicht nur eine fachliche Komponente, wie im vorherigen Abschnitt beschrieben. Um den Job gut ausführen zu können, sind auch eine Reihe von sozialen Fähigkeiten, die sogenannten *Softskills*, erforderlich. Die Arbeit erfordert es sehr häufig, Änderungen an vitalen Teilen der IT-Infrastruktur durchzuführen. Im geregelten

Fall bekommt man für Produktionssysteme Wartungsfenster, in denen Änderungen an Produktionsmaschinen durchgeführt werden dürfen. Oftmals ist es aber auch so, dass die Störung an einem Produktionssystem die Arbeit der Benutzer unmöglich macht. In diesem Fall sind Änderungen am Live-System erforderlich, und viele Administratoren sprechen hierbei von einer »Operation am offenen Herzen«.

Um das durchführen zu können, ist ein hohes Maß an **Selbstvertrauen** nötig, was somit auch gleich eine sehr wichtige Fähigkeit ist. Menschen mit geringem Selbstvertrauen sei gesagt, dass Selbstvertrauen – wie andere Fähigkeiten auch – trainierbar ist. Eine solide fachliche Basis kann dazu beitragen, das Selbstvertrauen zu unterstützen.

Selbstverantwortung und auch **Selbstdisziplin** sind zwei weitere wichtige Eigenschaften, die einen Systemadministrator auszeichnen. Mit Selbstverantwortung ist die Verantwortlichkeit für das eigene Handeln gemeint und die Fähigkeit, sich und anderen Fehler eingestehen zu können. Selbstdisziplin sorgt dafür, selbstständig Arbeit zu erledigen und auch in schwierigen Situationen, einem Regelwerk entsprechend, Produktionssysteme zu betreuen. Systemadministration ist Team sport. Daher sind im Umgang mit anderen Administratoren, Benutzern oder Vorgesetzten **Konfliktfähigkeit** und **Kooperation** ebenso wichtig wie die **Teamfähigkeit**.

Wie in den Job-Beschreibungen zu sehen ist, hat dieses Arbeitsfeld sehr viel mit **Kommunikation** zu tun. Hiermit ist sowohl mündliche wie auch schriftliche Kommunikation gemeint. Dazu gehört auch die Kenntnis der englischen Sprache, da der größte Teil der Fachliteratur in Englisch verfasst ist. Für den Fall, dass man mit dem Hersteller einer Software in Kontakt treten muss, ist es auch erforderlich, in Englisch korrespondieren zu können.

Das ist nicht zu unterschätzen. Anders als erwartet, beschäftigen sich Administratoren nicht als Selbstzweck mit den ihnen anvertrauten Maschinen. Was genau es damit auf sich hat, erklären wir in Abschnitt 1.4, »Unterbrechungsgesteuertes Arbeiten«.

1.2.2 Arbeitstechniken

Jetzt beschäftigen wir uns mit Fähigkeiten, die nicht direkt zu den Softskills zählen und auch nur wenig Bezug zu den fachlichen Fertigkeiten haben. Am Anfang der Arbeit steht der **Wille, zu verstehen**, um mit dem gewonnenen Wissen die anfallende Arbeit immer besser erledigen zu können. In letzter Konsequenz bedeutet das, dass ein Systemadministrator bereit ist, sein **Leben lang zu lernen** und sich ständig weiterzubilden – sei es durch Eigeninitiative oder durch Schulungen, die vom Arbeitgeber angeboten werden.

Damit einher geht das Streben, **Fehler nicht ein zweites Mal zu machen**. Das ist gerade im Dialog mit Vorgesetzten und Benutzern nicht zu unterschätzen. Gesucht werden dauerhafte Lösungen für Probleme und nicht Workarounds, die die Probleme verschieben. Der Neustart eines Computers behebt in der Regel kein Problem, er verschiebt nur die Auswirkungen.

Natürlich kann man sich bemühen, die Rechner in einem Rhythmus neu zu starten, sodass das Problem nicht mehr ans Tageslicht kommt, gelöst ist es damit aber nicht.

Für den Fall, dass man selbst nicht das Wissen hat, um ein Problem zu lösen oder um in einer entsprechend vorgegebenen Zeit eine Lösung zu finden, ist es elementar, dass man **Probleme abgeben kann**, beispielsweise an Kollegen oder auch an den externen Support des Herstellers einer Software oder Hardware. Häufig versuchen Systemadministratoren tage- oder sogar wochenlang, ein Problem selbst zu lösen, das durch die Inanspruchnahme des bereits bezahlten Supports in 30 Minuten gelöst wäre. »Sie haben für den Support bezahlt, also nutzen Sie ihn auch!«, wurde mir einmal von einem Vertriebsbeauftragten gesagt, und das hat mein Denken in dieser Hinsicht verändert.

Systemadministratoren sind in der Regel »Warmduscher« (oder »Beckenrandschwimmer« oder »Bergaufbremser« etc.), die es scheuen, unnötige Risiken einzugehen. Die Arbeitsmittel, die verwaltet werden, sind meist von zentraler Bedeutung für das Unternehmen, und da ist kein Platz für Cowboys. Ein Ausfall, der durch unsorgfältige Arbeit verursacht wird, kann neben finanziellem Schaden für das Unternehmen auch den eigenen Job kosten. Daher gilt es, genau und sorgfältig zu arbeiten und die Risiken richtig einzuschätzen. Wenn ein angebotener Dienst eingeschränkt oder gar nicht verfügbar ist, klingelt das Telefon im Regelfall pausenlos. In diesem Fall ist ein hohes Maß an **Stressresistenz** erforderlich, um selbst in diesen Situationen einen kühlen Kopf zu bewahren. Das heißt auch, professionell und freundlich im Umgang mit den aufgebrachten Benutzern zu sein und Fehler bei der Störungsbeseitigung zu vermeiden. Die so wichtige systematische und routinierte Herangehensweise an Probleme darf dem Druck nicht zum Opfer fallen, sonst schafft man damit mehr Probleme, die dann zusätzlich noch gelöst werden müssen.

Diese Form der Belastung wird auch *unterbrechungsgesteuertes Arbeiten* genannt. In diesem Arbeitsumfeld termingerecht Projektarbeiten zu erledigen und begonnene Aufgaben zu beenden, erfordert ein sehr hohes Maß an Disziplin und Selbstbeherrschung.

Eine der wichtigsten Eigenschaften, die ein Systemadministrator mitbringen muss, ist **Faulheit**. Ja, richtig gelesen! Die Faulheit bewirkt, dass man Fehler so behebt, dass sie endgültig gelöst sind. Ein hohes Maß an Automatisierung sorgt dafür, dass man Fehler nicht zweimal macht und dass man Arbeiten nicht mehrfach ausführen muss. Ein sinnvolles Monitoring von Prozessen und Diensten hilft Ihnen, Fehler zu erkennen, bevor Benutzer sie bemerken. Das steigert zum einen die Qualität der eigenen Arbeit und reduziert zum anderen die Anzahl der Nottelefonate, die geführt werden müssen.

Nicht zu vergessen ist, dass **Diskretion**, **Loyalität** und **Integrität** elementare Grundlagen für Arbeiten am Puls des Unternehmens darstellen. Diese Tugenden werden im letzten Abschnitt dieses Kapitels durch den Verhaltenskodex beschrieben, den sich die Systemadministratoren der SAGE selbst geben.

1.3 Das Verhältnis des Administrators zu Normalsterblichen

Wie am Anfang des Kapitels angedeutet wurde, fallen Systemadministratoren meist nicht auf, wenn der Betrieb gut läuft. Dass in der IT-Abteilung Menschen sitzen, die einen nahezu störungsfreien Betrieb der IT-Infrastruktur sicherstellen, wird meist erst dann bemerkt, wenn irgendetwas nicht funktioniert. Schlimmer noch: Eine landläufige Meinung besagt, dass Systemadministratoren meist diejenigen sind, die das teuer verdiente Geld des Unternehmens ausgeben und gar kein Geld einbringen. Das gilt natürlich nicht, wenn Sie bei Kunden Ihres Unternehmens Administrationsdienstleistungen erbringen.

Wenn Sie in einer größeren Firma mit einer eigenen Administrationsabteilung arbeiten, kennen Sie dieses Problem vermutlich nicht. Aber je kleiner die Firma ist, desto näher kommen Sie diesem Szenario.

1.3.1 Der Chef und andere Vorgesetzte

Oft ist es so, dass der Chef oder der direkte Vorgesetzte keinen fachlichen Hintergrund als Systemadministrator hat. Das muss er auch nicht, wenn es sein Job nicht erfordert. Dafür bringt er andere Fähigkeiten mit, die Sie für Ihre Arbeit nicht benötigen. Nicht alle Entscheidungen, die Sie als Systemadministrator betreffen, beruhen auf Kriterien, die Ihnen bekannt sind. Häufig spielen Partnerschaften mit anderen Unternehmen eine Rolle, und manchmal wird eine strategische Ausrichtung über mehrere Jahre festgelegt, und die nachträgliche Einflussnahme ist sehr begrenzt.

Daher ist es umso wichtiger, dass Sie mit Ihrem Chef richtig kommunizieren und Ihre Themen verständlich erläutern. Ein unterschiedliches Wissensniveau darf kein Grund sein, nicht respektvoll und professionell miteinander umzugehen. Bitte behalten Sie im Hinterkopf, dass die IT nicht in jedem Unternehmen das Kerngeschäft ist. Daher werden viele IT-Ausgaben als Geldausgaben ohne direkten Nutzen angesehen: »Warum brauchen Sie jetzt einen neuen Router, der bisherige funktioniert doch?« Bemühen Sie sich in jedem Fall, sachlich und kompetent zu informieren, sodass Ihr Anliegen verstanden wird.

Ihr Chef ist Ihnen gegenüber weisungsbefugt, er gibt Ihnen die Prioritäten vor. Die gesetzten Prioritäten können sich von Ihren eigenen elementar unterscheiden. Erklären Sie, warum und wie Sie die Prioritäten anders setzen würden. Die letzte Entscheidung liegt aber nicht bei Ihnen. Gerade dann, wenn Sie zu viele Aufgaben zu erledigen haben, ist das Positive daran, dass Sie durchaus auch Ihren Chef um Priorisierung bitten und ihn als Eskalationsstufe nutzen können.

Sie unterliegen einer Informationspflicht! Wenn Sie von rechtlichen Übertretungen oder schlimmstenfalls sogar von kriminellen Tätigkeiten erfahren, sind Sie verpflichtet, das unverzüglich Ihrem Vorgesetzten zu melden. Das heißt nicht, dass Ihr Vorgesetzter Sie zu kriminellen Handlungen zwingen darf.



1.3.2 Benutzer

»Unsere Systeme arbeiten am besten ohne Nutzer.«

Systemadministratoren verwalten die Werkzeuge oder die IT-Infrastruktur, die den Benutzern die Arbeit erleichtern oder ermöglichen. Bei allem Wissen, was dafür aufgewendet wird, sind es dennoch »nur« Werkzeuge. Benutzer sind mangels IT-Wissen keine Menschen zweiter Klasse. Sie kennen ihr Fachgebiet genauso gut wie der Administrator seines, und in der Regel sind sie es, die die »eigentliche« Arbeit im Unternehmen erledigen. Damit ist die Arbeit gemeint, mit der das Unternehmen Geld verdient.

Ebenso wie bei der Kommunikation mit dem Chef ist auch hier der Wissensunterschied kein Grund, nicht respektvoll mit dem Gegenüber umzugehen. Versuchen Sie, eine Sprache zu finden, die der Endanwender versteht. Hören Sie zu, und nehmen Sie die Anfragen und Probleme ernst.

Nutzer denken häufig, dass ihr Problem das wichtigste sei, weil es sie an der Erfüllung ihrer aktuellen Aufgabe hindert. Man kann ihnen aber auch erklären, dass Probleme, die alle im Unternehmen betreffen, Vorrang vor denen haben, die nur einige betreffen, und diese haben wiederum Vorrang vor den Problemen, die nur Einzelne betreffen. Ihr Chef kann das natürlich anders entscheiden. Sollte das nicht helfen, kann man als Systemadministrator immer noch freundlich und höflich bleiben und auf den Vorgesetzten verweisen, der die Bearbeitung dieser Anfrage priorisieren soll.

1.3.3 Andere Administratoren

Die Kommunikation unter Systemadministratoren ist – anders als bei Vorgesetzten oder Benutzern – sehr stark fachlich geprägt. Um schnell und effizient arbeiten zu können, müssen auch hier persönliche Befindlichkeiten außen vor bleiben. Das bedeutet insbesondere, dass andere Meinungen akzeptiert werden müssen. Diskussionen haben immer sachlich zu bleiben, denn nur so ist es möglich, das beste Resultat zu erzielen.

Niemand wird weniger akzeptiert, wenn er nachfragt. Es ist sogar deutlich besser, zu fragen und um Hilfe zu bitten, als mit Halbwissen zu versuchen, ein Problem zu lösen. Gegebenenfalls ist es besser, die Aufgabe abzugeben. Das heißt im Gegenzug aber auch, Aufgaben von Teammitgliedern zu übernehmen und ihnen mit dem eigenen Wissen zur Seite zu stehen, wenn sie Hilfe brauchen. Die Hilfe kann auch darin bestehen, alle Telefonanrufe anzunehmen, um den Kollegen zu entlasten und Freiräume zur Lösung der Probleme zu schaffen. Dieses Beispiel zeigt, dass die Unterstützung nicht nur fachlicher Natur sein muss. In Krisensituationen ist es notwendig, als Team zu funktionieren und nicht zu diskutieren.

Zu den wichtigen Aufgaben zählt es auch, Verfahren, Konfigurationen und bekannte Probleme und deren Lösungen zu dokumentieren, sodass keine Kopfmonopole existieren und im Fall, dass jemand Urlaub macht oder krank wird, die Arbeit erledigt werden kann.

1.4 Unterbrechungsgesteuertes Arbeiten

Die Arbeit als »klassischer Systemadministrator« teilt sich ganz grob in Projektarbeit und Störungsbehandlung auf. Projektarbeit erfordert eine längere Zeitphase, in der konzentriert an einem Stück gearbeitet werden kann. Dem steht die Störungsbehandlung entgegen, die meist auf Zuruf, per Telefon, SMS oder E-Mail eine direkte Aktion erwartet.

Wenn man sich um beide Aufgabengebiete zur gleichen Zeit kümmern muss, wird viel Zeit für das Umschalten benötigt. Normalerweise braucht man rund 15 Minuten nach einer Störung, um wieder auf dem Konzentrationslevel zu sein, den man vor der Störung hatte.

Sollte man mit mehreren Systemadministratoren im Team arbeiten, ist es hilfreich, einen Kollegen für die Störungsbeseitigung abzustellen, sodass die anderen in die Lage versetzt werden, möglichst unterbrechungsfrei Projektarbeit zu leisten.

Wenn das nicht möglich ist, ist es in jedem Fall hilfreich, sämtliche störenden Dienste abzuschalten, die nicht unbedingt zur Erfüllung der Aufgaben erforderlich sind. Zu diesen Diensten zählt alles, was auf dem Arbeitsrechner stören könnte, wie beispielsweise Instant Messenger, Chat-Systeme (IRC) oder E-Mail oder die entsprechenden Pendanten auf dem privaten Smartphone. Über welche Kanäle man erreichbar sein muss, entscheidet der Chef.

In jedem Fall hilft es aber, sich einen Plan über den Tagesablauf zu machen. Dieser Plan ist nicht starr, da er ja von äußeren Einflüssen abhängig ist.

Es sind oft mehr Aufgaben vorhanden, als der Arbeitstag Stunden hat. Daher möchten wir darauf hinweisen, dass Sie diese Planung auch mit Ihrem Vorgesetzten zusammen machen können. Sollten Sie in Rechtfertigungsnot kommen, was Sie den ganzen Tag getan haben, ist es hilfreich, sich über den Tag verteilt kurze Notizen zu machen, die neben der Uhrzeit ein Stichwort zur Störung oder zur geleisteten Arbeit enthalten.

Zeitplanung mit A.L.P.E.N.

Diese Methode der Tagesplanung geht auf Prof. Dr. Lothar J. Seiwert zurück (weitere Informationen finden Sie unter <https://lothar-seiwert.de>):

- ▶ **A:** Aufgaben und Termine schriftlich festhalten
- ▶ **L:** Länge der Bearbeitung realistisch schätzen
- ▶ **P:** Pufferzeiten (ca. 40 %) für Unvorhergesehenes
- ▶ **E:** Entscheiden, was wegfallen oder delegiert werden muss
- ▶ **N:** Nachkontrolle der Einschätzung im Rückblick

Ob das Verfahren für Sie funktioniert, müssen Sie in Ihrem Umfeld testen.

1.5 Einordnung der Systemadministration

Welche Rolle der Administrator in Unternehmen spielt, hängt grundsätzlich von der Größe des Unternehmens und insbesondere der IT-Abteilung ab. Vom »Mädchen für alles« bis hin zum sehr auf ein Fachgebiet spezialisierten Experten gibt es eine sehr große Bandbreite.

1.5.1 Arbeitsgebiete

In größeren Unternehmen sind die Rollen, die sich in der Systemadministration finden, nämlich Architektur, Engineering und Operation, auf einzelne Teams verteilt. In kleineren Unternehmen vereint der Systemadministrator alle Rollen auf sich. Zunächst eine Begriffsklärung zur Unterscheidung:

Architekten gestalten Lösungen und übergeben sie **System Engineers** zur Implementierung, die zusätzlich noch Betriebspläne und andere Konzepte schreiben. **Operatoren** sorgen für den Betrieb von Systemen. Die Grenzen sind, gerade in kleineren Unternehmen, fließend. Umgangssprachlich lässt sich das auch so zusammenfassen: Architekten malen bunte Bilder mit konkreten Wunschvorstellungen. Engineers prüfen diese auf Praxistauglichkeit und implementieren sie. Operatoren baden das aus, was sich Architekten und Engineers ausgedacht haben. Idealerweise gibt es Rückkopplungen zwischen den einzelnen Teilbereichen. Gerade weil die Grenzen fließend sind, kann es sein, dass die Ausprägung je nach Unternehmen unterschiedlich ist. Häufig ist es so, dass Systemoperatoren 1st- oder 2nd-Level-Support leisten und System Engineers als Stufe dahinter 2nd- bzw. 3rd-Level-Support bieten.

Neue Arbeitskulturen – wie beispielsweise DevOps (siehe Abschnitt 1.5.2) – sorgen für eine Neuausrichtung und ein neues Rollenkonzept. Da ist immer noch einiges in Bewegung, auch die generelle Unterscheidung zwischen *Systembetrieb* (oder Systemtechnik bzw. Systemintegration) und *Anwendungsentwicklung* ändert sich. (Amazon, Facebook und Google machen das vor, und das, was sie herausfinden, »tröpfelt« auch in kleinere Umgebungen.) Wer in kleinen Umgebungen arbeitet, bekommt von diesen Konzepten nichts mit, da gibt es zumeist »Mädchen für alles« und keine weitere Unterscheidung.

Architektur

In der Regel ist der Startpunkt für neue Installationen oder Umgebungen ein Kundenauftrag, eine strategische Entscheidung oder schlicht der Wunsch nach etwas Neuem. Das wird im Normalfall durch ein Projekt abgebildet. Im Folgenden verwenden wir die englischen Fachbegriffe und konzentrieren uns nur auf den für die Systemadministration relevanten Teil. Was Projekte sind und wie diese verwaltet werden, können andere Personen besser beschreiben. In einer ersten Phase setzt sich ein Architekturteam zusammen und beschreibt die Lösung auf zwei Ebenen, die für die Systemadministration relevant sind: ein Gesamtbild – das »Big Picture« – der Lösung und eine Infrastruktur-Sicht (da finden wir uns wieder). Es ist hilfreich, das große Bild zu kennen, um Entscheidungen zu verstehen. Beteiligt sind »Solu-

tion Design« bzw. »Solution Architecture« und »Infrastructure Design« bzw. »Infrastructure Architecture«.

In der nächsten Phase kümmern sich »Technical Architects« um die Umsetzung der Ideen. Es wird ein konzeptionelles Bild auf Ebene der jeweiligen Technologie in den Feldern System, Database, Middleware, Software und Infrastructure gebaut und eine Machbarkeitsstudie (»Proof of concept«) entwickelt. Damit wird grundlegend gezeigt, dass die Bilder, die die Architekten malen, auch tatsächlich umsetzbar sind. »Technical Architects« oder »Domain Architects« werden in manchen Firmen auch »Technical Solution Engineers« genannt und haben eine Brückenstellung zwischen Architektur und Engineering inne.

Engineering

System Engineers bauen Testsysteme und erstellen Operating System Builds, die später betrieben werden können. Diese Builds werden auch benutzt, um systemseitig Patches und Upgrades zu testen. Der Sinn dahinter ist, den Aufbau eines Systems nachvollziehbar und wiederholbar zu halten sowie die Größenordnung der verwendeten Systeme sicherzustellen.

Der Aufbau von Werkzeugkästen, die den betreibenden Administratoren später im Betrieb nützlich sind, gehört ebenfalls zu den Aufgaben, und in manchen Umgebungen werden diese Hilfssysteme (Beispiele sind Imageserver und Konfigurationsmanagementsysteme wie *Puppet* oder *Chef*) sogar vom Engineering betrieben. Dieses Engineering findet wie die technische Architektur in den gleichen Feldern statt: System, Database, Middleware, Software und Infrastructure. Der für viele unangenehme Teil der Arbeit ist das Schreiben von Betriebskonzepten, häufig in englischer Sprache, und Betriebshandbüchern. Hinzu kommt die weitere Pflege dieser Dokumentation. Am Ende der Projektphase werden die Systeme dem Betrieb übergeben und von »Betrieblern« verwaltet.

Operation

Der »Lebenslauf eines Systems« dient hier als Beispiel für den Betrieb, um Konzepte zu verdeutlichen. Zunächst müssen die Systeme gekauft werden. In größeren Umgebungen sorgt dafür eine Einkaufsabteilung (Purchase Department). Die Vorgaben kommen von der Architektur und werden durch das Engineering geprüft. Eine Provisionierungsabteilung (= Aufbau der Geräte nach Vorgaben des Engineerings) baut die Systeme auf, montiert sie ins Rack, verkabelt sie, installiert das Basissystem, konfiguriert das Netzwerk und bindet sie ans LDAP (oder Active Directory, NIS etc.) an.

Anschließend erfolgt das »Customizing« bzw. die Anpassung an die spezifische Aufgabe. Das wird in manchen Firmen durch das Engineering erledigt, im Normalfall (die Vorgaben sind ja dokumentiert) durch das Provisioning und später im Betrieb explizit durch die Systemadministratoren bzw. das Operation-Team. Systemadministration bzw. Operation ist verantwortlich für den kompletten Betrieb des Servers und leistet 1st-Level-Support. Das sind insbesondere die drei Felder Incident Management, Problem Management und Change Management.

Zum Ende der Lebenszeit der Systeme werden diese durch ein (End-of-life-)Provisioning-Team heruntergefahren, ausgebaut und entsorgt. Im Betrieb gibt es natürlich auch eine Schnittstelle zum Engineering und (fast noch wichtiger) eine Feedback-Schleife. Engineering-Teams leisten häufig 2nd-Level-Support und helfen weiter, wenn der 1st Level an seine Grenzen gekommen ist. Neben Incident Management wird vor allem beim Problem Management (oder bei Taskforces) unterstützt. Das, was dort herausgefunden wird, fließt wieder in die weiter oben angeführten Builds sowie in die Architektur ein. Die im Betrieb gesammelte Erfahrung, insbesondere was die Funktionsfähigkeit angeht, muss Einfluss auf die Produkte und Designs haben. Nur der Vollständigkeit halber sei erwähnt, dass der Hersteller meistens den 3rd Level besetzt; die Erfahrungen, die dort gemacht werden, müssen natürlich Einfluss auf Betrieb, Engineering, Architecture und Design haben.

Für diejenigen, die sich tiefer in das Thema einarbeiten wollen, ist ITIL (*IT Infrastructure Library*) – <https://de.wikipedia.org/wiki/ITIL> – das passende Stichwort. ITIL beinhaltet eine Serie an bewährten Verfahren, die man im IT-Service-Management verwenden kann. ITIL-Wissen zu haben, schadet niemandem, der in der IT arbeitet. Viele Arbeitgeber fordern es sogar ein.

1.5.2 DevOps

Nicht ganz so neu, wie gemeinhin behauptet wird, ist eine Strömung (in Bezug auf Systemadministration), die sich *DevOps* nennt. Man könnte DevOps auch als Kultur begreifen. Grundsätzlich gibt es dabei zwei verschiedene Definitionen, die den Begriff DevOps beschreiben. Die erste und weiter verbreitete Definition beschreibt DevOps als eine Mischung aus agiler Systemadministration (siehe Kasten »Agiles Manifest«, bezogen auf Softwareentwicklung) und Zusammenarbeit mit Entwicklern, um gemeinsam an dem Ziel zu arbeiten, höhere Kundenzufriedenheit herzustellen.

In der Wikipedia (https://de.wikipedia.org/wiki/Agile_Softwareentwicklung#Werte) findet sich das im Kasten dargestellte »Agile Manifest«.

Agiles Manifest

»Wir erschließen bessere Wege, Software zu entwickeln, indem wir es selbst tun und anderen dabei helfen. Durch diese Tätigkeit haben wir diese Werte zu schätzen gelernt:

- ▶ **Individuen und Interaktionen** mehr als Prozesse und Werkzeuge
- ▶ **Funktionierende Software** mehr als umfassende Dokumentation
- ▶ **Zusammenarbeit mit dem Kunden** mehr als Vertragsverhandlung
- ▶ **Reagieren auf Veränderung** mehr als das Befolgen eines Plans

Das heißt, obwohl wir die Werte auf der rechten Seite wichtig finden, schätzen wir die Werte auf der linken Seite höher ein.«

Einen Schritt weiter gedacht, führt DevOps dazu, dass *Operations* (der Betrieb von Systemen) und *Development* (die Weiterentwicklung von Systemen) verschmelzen. Ein großer Vertreter dieser Art des Betriebs ist die Suchmaschine Google, die diesen Berufszweig *Site Reliability Engineering* nennt. Solange ein Produkt noch nicht in Produktion ist, wird es einzig und allein von Entwicklern nach vorn getrieben. Bei Produktionsreife wird nicht nur die Software an den Betrieb übergeben, sondern auch die Weiterentwicklung des Produkts. Dieses Prinzip löst in sehr großen Unternehmen, in denen eine Vielzahl an Systemen für eine Aufgabe bereitsteht – beispielsweise bei Cloud-Diensten –, die klassische Systemadministration ab.

Die zweite Definition bezieht sich auf den hohen Automatisierungsgrad, den Administratoren mithilfe von selbst geschriebenen Programmen und Skripten erreichen können und sollen. Als Motto gilt hier: **Lasst uns damit aufhören, die Arbeit der Computer zu tun!**

Das bedeutet insbesondere, dass Produktionsprobleme mithilfe von Software (automatisiert) gelöst werden. Der »Klassiker« in diesem Umfeld ist die skriptgesteuerte Bereitstellung von neuen Systemen, um die Zeit von Auftragseingang bis Auftrags erledigung so gering wie möglich zu halten. Für jemanden, der nur ein System pro Jahr installieren muss, erschließt sich der Nutzen nicht, was verständlich ist.

Alle anderen werden verstehen, dass mit der Automatisierung eine Möglichkeit existiert, zu wiederholbaren Ergebnissen zu kommen. Wenn Fehler im Nachhinein bei einer Vielzahl von Systemen auftreten, ist klar, dass die Ursache in der Bereitstellung der Systeme zu finden ist und die Fehler auch dort zu beheben sind. Nachfolgende Installationen werden diese Probleme nicht mehr haben. Vielleicht lässt sich ein DevOps-Engineer am ehesten als *Systementwickler* beschreiben, der dafür sorgt, dass es niemanden mehr gibt, der Systeme betreiben muss, weil sie sich autonom, durch die geschriebene Software, selbst verwalten. Ein sehr sehenswerter – englischsprachiger – Vortrag mit dem Thema »PostOps: A Non-Surgical Tale of Software, Fragility, and Reliability« findet sich unter der folgenden URL:

<https://www.usenix.org/conference/lisa13/technical-sessions/plenary/underwood>

Durchgesetzt hat sich letzten Endes die folgende Definition, die in englischer Sprache unter der URL <https://www.itskeptc.org/content/define-devops.html> zu finden ist.

Frei übersetzt:

DevOps ist die agile Bereitstellung von IT-Diensten, die benötigt wird, um mit dem Rhythmus der agilen IT-Entwicklung mitzuhalten.

*DevOps ist eine Philosophie und weder eine Methode noch ein Framework, eine Wissenssammlung oder *schauder* irgendein Hilfsmittel von einer Firma.*

DevOps ist die Philosophie, Entwicklung und Betrieb in Bezug auf Kultur, Praxis und Werkzeuge zu vereinen, um eine beschleunigte und häufigere Bereitstellung von Änderungen in der Produktion zu erreichen.

Und wie jeder Kultur- und Paradigmenwechsel ist auch DevOps ein schmerzhafter Prozess.

1.6 Ethischer Verhaltenskodex

Die bereits angesprochene *System Administrators Guild* wurde aufgelöst und ist neu in der *LISA* aufgegangen. *LISA* ist die *Special Interest Group for System Administrators* innerhalb der *Usenix Association*. Den Verhaltenskodex der *LISA*, den *System Administrators' Code of Ethics*, nach dem sich die Mitglieder freiwillig richten, finden Sie unter der URL <https://www.usenix.org/system-administrators-code-ethics>.

Im Folgenden lesen Sie die deutsche Übersetzung dieses Kodex, dem sich die Vereinigungen *LISA*, *USENIX (The Advanced Computing Systems Association)* und *LOPSA (League of Professional System Administrators)* verpflichten:

Dem »Ethischen Verhaltenskodex für Systemadministratoren« verpflichten wir uns als professionelle Systemadministratoren, um den höchsten Anforderungen an ethisches und professionelles Verhalten gerecht zu werden, und wir stimmen zu, uns vom Verhaltenskodex leiten zu lassen und jeden Systemadministrator zu ermutigen, dasselbe zu tun.

Professionalität

- ▶ Ich behalte am Arbeitsplatz standesgemäßes Verhalten bei und lasse nicht zu, dass persönliche Gefühle oder Überzeugungen mich dazu verleiten, Mitmenschen unfair oder unprofessionell zu behandeln.

Persönliche Integrität

- ▶ Im fachlichen Umgang bin ich ehrlich und spreche offen über meine Fähigkeiten und die Auswirkungen meiner Fehler. Ich frage andere um Hilfe, wenn es notwendig ist.
- ▶ Ich vermeide Interessenkonflikte und Voreingenommenheit, wann immer es möglich ist. Im Falle einer Konsultation werde ich, wenn ich einen Interessenkonflikt habe oder voreingenommen bin, das entsprechend kundtun und falls notwendig die Anfrage wegen Befangenheit ablehnen.

Privatsphäre

- ▶ Ich erlaube mir Zugang zu privaten Informationen auf Computersystemen nur, wenn es im Zuge meiner fachlichen Pflichten notwendig wird. Ich werde meine Schweigepflicht in Bezug auf alle Informationen, zu denen ich Zugang habe, aufrechterhalten und wahren, egal wie ich zu diesem Wissen gekommen bin.

Gesetze und Richtlinien

- ▶ Ich werde mich und andere über relevante Gesetze, Vorschriften und Richtlinien bezüglich der Erfüllung meiner Pflichten unterrichten.

Kommunikation

- ▶ Ich werde mich mit dem Management sowie den Benutzern und Kollegen über Computerangelegenheiten von beiderseitigem Interesse verständigen. Ich bemühe mich, zuzuhören und die Bedürfnisse aller Personen zu verstehen.

Systemintegrität

- ▶ Ich werde bestrebt sein, die nötige Integrität, Zuverlässigkeit und Verfügbarkeit der Systeme sicherzustellen, für die ich verantwortlich bin.
- ▶ Ich werde jedes System in einer Art und Weise entwerfen und pflegen, dass es den Nutzen für die Organisation unterstützt.

Ausbildung

- ▶ Ich werde mein Fachwissen und meine beruflichen Fähigkeiten kontinuierlich aktualisieren und erweitern. Darüber hinaus werde ich mein Wissen und meine Erfahrungen mit anderen teilen.

Verantwortung gegenüber der Computing Community

- ▶ Ich werde mit der großen Computing Community kooperieren, um die Integrität von Netzwerk- und IT-Ressourcen sicherzustellen.

Soziale Verantwortung

- ▶ Als informierter Fachmann werde ich das Schreiben und das Übernehmen von relevanten Grundsätzen und Gesetzen unterstützen, die mit diesen ethischen Prinzipien einhergehen.

Ethische Verantwortung

- ▶ Ich werde mich bemühen, einen sicheren, gesunden und produktiven Arbeitsplatz zu schaffen und beizubehalten.
- ▶ Ich werde mein Bestes geben, um Entscheidungen zu treffen, die mit der Sicherheit, Privatsphäre und dem Wohlergehen meiner Umgebung und der Öffentlichkeit vereinbar sind, und Faktoren unverzüglich ausschließen, die unvorhersehbare Risiken oder Gefahren darstellen. Ich werde ehrlich gemeinte Kritik an fachlicher Arbeit wenn nötig geben und akzeptieren und werde Beiträge von anderen korrekt anerkennen.
- ▶ Ich werde durch mein Vorbild führen, einen hohen ethischen Anspruch und ein hohes Maß an Leistung in allen meinen Aufgaben beibehalten. Ich werde Arbeitskollegen und Mitarbeiter beim Einhalten dieses Verhaltenskodexes unterstützen.

1.7 Administration – eine Lebenseinstellung?

Zugegebenermaßen ist diese Überschrift ein wenig reißerisch, aber sie verdeutlicht, dass hinter der Administration mehr steckt, als es auf den ersten Blick den Anschein hat.

Wie in Abschnitt 1.2.2 über die Arbeitstechniken beschrieben, ist der »Wille, zu verstehen« elementar für die Arbeit als Systemadministrator. Dieser Wille beschränkt sich aber nicht nur auf die fachlichen Aufgaben, die der Beruf mit sich bringt, sondern auch auf andere Bereiche des Lebens, und damit kommen wir zur Lebenseinstellung. Viele der im genannten

Abschnitt vorgestellten Fähigkeiten und Fertigkeiten entfalten ihre Wirkung auch im täglichen Leben. Loyalität und Integrität sind selbstverständlich nicht nur auf das professionelle Leben beschränkt. Verlässlichkeit ist eine Charaktereigenschaft, die nahezu überall geschätzt wird.

Es wird in vielen Bereichen über das lebenslange Lernen gesprochen. Damit sind bezogen auf die Systemadministration nicht nur neue Versionen von Betriebssystemen und anderen Programmen gemeint, sondern auch die Weiterbildung hinsichtlich neuer Techniken und neuer Verfahrensweisen, wie man seine Arbeit besser strukturieren und ausführen kann.

Wer es schafft, unter großem Druck und viel Stress noch verwertbare Ergebnisse im Beruf zu erzielen, der kann diese Fähigkeiten auch einsetzen, um sein eigenes Leben zu strukturieren. Damit ist dann auch schon fast alles gesagt:

Ich bin Systemadministrator.

TEIL I
Grundlagen

Kapitel 2

Der Bootvorgang

Der Startvorgang eines Linux-Systems ist die Basis dafür, überhaupt etwas mit dem System anfangen zu können. Wir geben einen Einblick in den Bootloader und die initiale Ramdisk. Wir widmen uns init-Skripten und blicken auf »event-gesteuertes Starten« mittels »systemd«.

Mit dem Bootloader wird das Betriebssystem gestartet. Nachdem das BIOS den mehr oder weniger ausführlichen Systemcheck durchgeführt hat, werden die Bootmedien in der Reihenfolge der Präferenzen abgearbeitet. Wenn es zur Festplatte oder analog zur SSD oder NVMe kommt, werden die ersten 512 Byte der Festplatte ausgewertet; in diesen ist der *Master Boot Record* (MBR) zu finden. Von den 512 Byte sind die ersten 446 für den Bootloader reserviert. In diesem begrenzten Bereich lassen sich keine großen Programme unterbringen, daher wird der Bereich dafür genutzt, Code von anderer Stelle nachzuladen.

Der frühere *Linux Loader* (LILO) ist heute kaum noch verbreitet, daher beschränken wir uns im Weiteren auf die Weiterentwicklung des *Grand Unified Bootloader* (GRUB) mit dem Namen *GRUB 2*.

2.1 Der Bootloader GRUB 2

Mit *GRUB 2* wurde GRUB von Grund auf neu entwickelt. Die Entwickler haben sich sehr viel Zeit gelassen und sich in kleinen Sprüngen der Version 2 genähert. GRUB2 ist bei allen in diesem Buch verwendeten Distributionen in der Version 2.06 enthalten.

Da die Macher des Bootloaders einen sehr konservativen Ansatz bei der Versionierung verfolgen, darf man sich generell nicht von dem Begriff »beta« schrecken lassen. Die erste Version von GRUB hat beispielsweise nie die Version 1 erreicht; die höchste Versionsnummer war 0.97.



2.1.1 Funktionsweise

Der große Unterschied von GRUB 2 im Vergleich zu GRUB ist, dass die ehemaligen Stages 1.5 und 2, vom Laden der Dateisystemtreiber bis zum Anzeigen des Bootmenüs, zu einem einzigen Stage 2 zusammengelegt wurden. Dabei nutzt GRUB 2 einen minimalistischen und

sehr kleinen Kern und viele Module, die je nach Bedarf nachgeladen werden können, um auf die Konfigurationsdatei zugreifen zu können. Auf diese Weise unterstützt GRUB 2 auch das Starten von LVM oder Software-RAIDs mit *md*.

2.1.2 Installation

GRUB 2 wird genauso wie GRUB mit `grub-install` (bei CentOS und openSUSE mit `grub2-install`) installiert, allerdings müssen Sie bei GRUB 2 angeben, wo der Bootloader installiert werden soll. Dabei zeigt GRUB 2 deutlich weniger Ausgaben bei der Installation (siehe Listing 2.1):

```
# Debian und Ubuntu
# grub-install /dev/sda
Installing for i386-pc platform.
Installation finished. No error reported.

# CentOS und openSUSE, mit "2" hinter grub
# grub2-install /dev/sda
Installing for i386-pc platform.
Installation finished. No error reported.
```

Listing 2.1 Installation von »GRUB 2«

2.1.3 Konfiguration

Die Konfigurationsdatei von GRUB 2 liegt in `/boot/grub/grub.cfg` bzw. `/boot/grub2/grub.cfg`. Bitte ändern Sie diese Datei nicht von Hand, sie wird von den Skripten unter `/etc/grub.d` erstellt. In diesem Verzeichnis wird den Skripten eine Nummer vorangestellt, um die Reihenfolge festzulegen. Das Verfahren, die Konfiguration aus einzelnen Bausteinen (Skripten) zusammenstellen zu lassen, macht GRUB 2 deutlich flexibler und besser automatisierbar als seinen Vorgänger: So werden installierte Kernel automatisch erkannt und in das Bootmenü aufgenommen. Die hohe Flexibilität wird allerdings durch eine komplexere Konfiguration erkauft. Ohne gutes Shell-Scripting-Know-how kommt man da nicht viel weiter.

Einfachere Konfigurationen wie das Bootmenü sind relativ leicht machbar. Einstellungen, die das komplette Bootverhalten beeinflussen, wie beispielsweise Timeouts oder der Kernel, der standardmäßig gestartet werden sollte, werden in der Datei `/etc/default/grub` vorgenommen. In der von uns beschriebenen Ubuntu-Version 20.04 sind die folgenden Dateien im Verzeichnis `/etc/grub.d` zu finden:

► `00_header`

Mit diesem Skript werden die Standardeinstellungen aus der Datei `/etc/default/grub` gesetzt.

- ▶ **05_debian_theme**
Diese Datei sorgt für das Aussehen des Bootmenüs: Hier werden Farben und Hintergrundbild definiert.
- ▶ **10_linux**
Dieses Skript nimmt alle installierten Kernel in das Bootmenü auf.
- ▶ **10_linux_zfs**
Mit diesem Skript werden die ZPools von ZFS initialisiert.
- ▶ **20_linux_xen**
Hier werden besondere Einstellungen vorgenommen und spezielle Kernel für die Xen-Virtualisierung gesetzt.
- ▶ **30_os-prober**
Dieses Skript sucht nach installierten (anderen) Betriebssystemen und nimmt sie in das Bootmenü auf.
- ▶ **30_uefi-firmware**
Besondere Einstellungen für *UEFI-Systeme* werden mit diesem Skript getroffen.
- ▶ **40_custom**
Diese Datei ist für eigene Booteinträge vorhanden.
- ▶ **41_custom**
Hiermit wird die */boot/grub/custom.cfg* eingebunden, sofern sie existiert.
- ▶ **README**
Diese Datei enthält Hintergrundinformationen für die Skripte in diesem Verzeichnis.

Die Skriptnummern, die mit 00, 10 oder 20 beginnen, sind reserviert. Alle Nummern dazwischen können Sie für eigene Skripte verwenden. Je nachdem, welche Nummer Sie Ihrem Skript geben, wird es früher oder später im Prozess ausgeführt. Apropos »ausgeführt«: Die Skripte unterhalb von */etc/grub.d* müssen alle ausführbar sein.

Wir legen jetzt einen neuen Eintrag im Bootmenü an. Dazu werden am Ende der Datei *40_custom* die Zeilen aus Listing 2.2 neu eingefügt:

```
#!/bin/sh
exec tail -n +3 $0
# This file provides an easy way to add custom menu entries.  Simply type the
# menu entries you want to add after this comment.  Be careful not to change
# the 'exec tail' line above.
menuentry "Ubuntu 20.04.2 LTS, kernel 5.4.0-65-Adminbuch" {
    set root='(hd0,1)'
    linux /vmlinuz-5.4.0-65-generic root=/dev/mapper/ubuntu--vg-root ro console=hvc0
    initrd /initrd.img-5.4.0-65-generic
}
```

Listing 2.2 Eigener Eintrag in der Datei »40_custom«

Das Skript sorgt nur dafür, dass die Zeilen ab der dritten Zeile ausgegeben werden. Die eigentliche Konfiguration findet sich in der geschweiften Klammer nach dem `menuentry`, der den Text des Eintrags im Bootmenü enthält.

Wie gewohnt kennzeichnet `set root` die Partition, in der sich das Verzeichnis `/boot` befindet. Natürlich bietet GRUB 2 eine Besonderheit: Die Festplattennummerierung beginnt bei 0, und die Nummerierung der Partition beginnt bei 1. So wird aus der Partition `/dev/sdb3` unter GRUB `hd1,2` und unter GRUB 2 `hd1,3`.

Nach `linux` (früher `kernel`) folgt der zu startende Betriebssystemkern. Und `initrd` ist so, wie bereits beschrieben, die Initial Ramdisk. Mithilfe von `update-grub`, siehe Listing 2.5 wird ein neuer Bootloader geschrieben, und beim nächsten Start finden wir unseren neuen Eintrag im Bootmenü.



Wie bereits beschrieben, ist GRUB 2 modular aufgebaut und bringt keine Treiber mit, daher muss man eventuell noch Module mit dem Kommando `insmod` hinzuladen, um aus einem einfachen Menüeintrag ein startfähiges System zu machen.

Beispiele dafür sind LVM, besondere Dateisysteme oder auch RAID. Alle verfügbaren Module Ihrer GRUB-2-Installation finden sich im Verzeichnis `/boot/grub/i386-pc` oder `/boot/grub2/i386-pc` und enden auf `.mod`. In Listing 2.3 finden Sie die Module eines Debian-Stretch-Systems:

```
root@debian:~# ls /boot/grub/i386-pc
915resolution.mod    gcry_whirlpool.mod    password_pbkdf2.mod
acpi.mod             gdb.mod               pata.mod
adler32.mod          geli.mod              pbkdf2.mod
affs.mod             gettext.mod           pbkdf2_test.mod
afs.mod             gfxmenu.mod           pci.mod
ahci.mod            gfxterm.mod           pcidump.mod
all_video.mod        gfxterm_background.mod  plan9.mod
aout.mod            gfxterm_menu.mod      play.mod
archelp.mod          gptsync.mod           png.mod
at_keyboard.mod      gzio.mod              priority_queue.mod
ata.mod             halt.mod              probe.mod
backtrace.mod        hashsum.mod           procfs.mod
bfs.mod             hdparm.mod            progress.mod
biosdisk.mod         hello.mod              pxe.mod
bitmap.mod           help.mod              pxechain.mod
bitmap_scale.mod     hexdump.mod           raid5rec.mod
blocklist.mod        hfs.mod               raid6rec.mod
boot.img             hfsplus.mod           read.mod
boot.mod             hfspluscomp.mod       reboot.mod
bsd.mod             http.mod              regexp.mod
btrfs.mod           hwmatch.mod           reiserfs.mod
```

bufio.mod	iorw.mod	relocator.mod
cat.mod	iso9660.mod	romfs.mod
cbfs.mod	jfs.mod	scsi.mod
cbls.mod	jpeg.mod	search.mod
cbmemc.mod	keylayouts.mod	search_fs_file.mod
cbtable.mod	keystatus.mod	search_fs_uuid.mod
cbtime.mod	ldm.mod	search_label.mod
chain.mod	legacy_password_test.mod	sendkey.mod
cmdline_cat_test.mod	legacycfg.mod	serial.mod
cmosdump.mod	linux.mod	setjmp.mod
cmostest.mod	linux16.mod	setjmp_test.mod
cmp.mod	loadenv.mod	setpci.mod
command.lst	loopback.mod	sfs.mod
configfile.mod	ls.mod	signature_test.mod
core.img	lsacpi.mod	sleep.mod
cpio.mod	lsapm.mod	sleep_test.mod
cpio_be.mod	lsmmap.mod	spkmodem.mod
cpuid.mod	lspci.mod	squash4.mod
crc64.mod	luks.mod	syslinuxcfg.mod
crypto.lst	lvm.mod	tar.mod
crypto.mod	lzopio.mod	terminal.lst
cryptodisk.mod	macbless.mod	terminal.mod
cs5536.mod	macho.mod	terminfo.mod
date.mod	mda_text.mod	test.mod
datehook.mod	mdraid09.mod	test_blockarg.mod
datetime.mod	mdraid09_be.mod	testload.mod
disk.mod	mdraid1x.mod	testspeed.mod
diskfilter.mod	memdisk.mod	tftp.mod
div_test.mod	memrw.mod	tga.mod
dm_nv.mod	minicmd.mod	time.mod
drivemap.mod	minix.mod	tr.mod
echo.mod	minix2.mod	trig.mod
efiemu.mod	minix2_be.mod	true.mod
efiemu32.o	minix3.mod	truecrypt.mod
efiemu64.o	minix3_be.mod	udf.mod
ehci.mod	minix_be.mod	ufs1.mod
elf.mod	mmap.mod	ufs1_be.mod
eval.mod	moddep.lst	ufs2.mod
exfat.mod	modinfo.sh	uhci.mod
exfctest.mod	morse.mod	usb.mod
ext2.mod	mpi.mod	usb_keyboard.mod
extcmd.mod	msdospart.mod	usbms.mod
fat.mod	multiboot.mod	usbserial_common.mod

file.mod	multiboot2.mod	usbserial_ftdi.mod
font.mod	natedisk.mod	usbserial_pl2303.mod
freedos.mod	net.mod	usbserial_usbdebug.mod
fs.lst	newc.mod	usbtest.mod
fshelp.mod	nilfs2.mod	vbe.mod
functional_test.mod	normal.mod	verify.mod
gcry_arcfour.mod	ntfs.mod	vga.mod
gcry_blowfish.mod	ntfscomp.mod	vga_text.mod
gcry_camellia.mod	ntldr.mod	video.lst
gcry_cast5.mod	odc.mod	video.mod
gcry_crc.mod	offsetio.mod	video_bochs.mod
gcry_des.mod	ohci.mod	video_cirrus.mod
gcry_dsa.mod	part_acorn.mod	video_colors.mod
gcry_idea.mod	part_amiga.mod	video_fb.mod
gcry_md4.mod	part_apple.mod	videoinfo.mod
gcry_md5.mod	part_bsd.mod	videotest.mod
gcry_rfc2268.mod	part_dfly.mod	videotest_checksum.mod
gcry_rijndael.mod	part_dvh.mod	xfs.mod
gcry_rmd160.mod	part_gpt.mod	xnu.mod
gcry_rsa.mod	part_msdos.mod	xnu_uuid.mod
gcry_seed.mod	part_plan.mod	xnu_uuid_test.mod
gcry_serpent.mod	part_sun.mod	xzio.mod
gcry_sha1.mod	part_sunpc.mod	zfs.mod
gcry_sha256.mod	partmap.lst	zfsencrypt.mod
gcry_sha512.mod	parttool.lst	zfsinfo.mod
gcry_tiger.mod	parttool.mod	
gcry_twofish.mod	password.mod	

Listing 2.3 GRUB-2-Module eines Debian-Stretch-Systems

Auf dem gleichen System findet sich in der `/boot/grub/grub.cfg` ein Beispiel dafür, wie ein Teil dieser Module eingesetzt wird (siehe Listing 2.4):

```
[...]  
menuentry 'Debian GNU/Linux, with Linux 3.16.0-4-amd64' --class debian \  
--class gnu-linux --class gnu --class os $menuentry_id_option \  
'gnulinux-3.16.0-4-amd64-advanced-eac6da17-314e-43c0-956f-379457a505fa' {  
    load_video  
    insmod gzio  
    if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi  
    insmod part_msdos  
    insmod ext2  
    set root='hd0,msdos1'  
    if [ x$feature_platform_search_hint = xy ]; then
```



```

search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 \
--hint-efi=hd0,msdos1 --hint-baremetal=ahci0,msdos1 \
eac6da17-314e-43c0-956f-379457a505fa
else
search --no-floppy --fs-uuid --set=root eac6da17-314e-43c0-956f-379457a505fa
fi
echo 'Loading Linux 3.16.0-4-amd64 ...'
linux /boot/vmlinuz-3.16.0-4-amd64 root=UUID=eac6da17-314e-43c0-956f-\
379457a505fa ro quiet
echo 'Loading initial ramdisk ...'
initrd /boot/initrd.img-3.16.0-4-amd64
}
[...]
```

Listing 2.4 Die Optionen des Standardkernels aus der »/boot/grub/grub.cfg«

Änderungen in der Datei */boot/grub/grub.cfg* werden nicht automatisch übernommen. Mit dem Kommando `update-grub` wird GRUB 2 aktualisiert, wie in Listing 2.5 zu sehen ist:



```

root@debian:~# update-grub
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.16.0-4-amd64
Found initrd image: /boot/initrd.img-3.16.0-4-amd64
done
```

Listing 2.5 »update-grub«

Interessant ist, dass die Konfigurationsdatei */etc/default/grub* ein Shell-Skript ist. Dort werden nur Variablen gesetzt, die nach dem Aufruf von `update-grub` durch */etc/grub.d/00_header* ausgewertet werden. In der folgenden Auflistung finden Sie die wichtigsten Variablen:

- ▶ **GRUB_DEFAULT=0**
Hiermit wird der Standardeintrag gesetzt.
- ▶ **GRUB_TIMEOUT=5**
Nach Ablauf der durch `TIMEOUT` gesetzten Zeit wird der Standardeintrag gestartet.
- ▶ **GRUB_HIDDEN_TIMEOUT=0**
Wenn nur ein Betriebssystem existiert, wird dieser Wert als Wartezeit benutzt. Sobald ein weiterer Eintrag hinzukommt, ist der Wert bedeutungslos.
- ▶ **GRUB_HIDDEN_TIMEOUT_QUIET=true**
Mit `true` wird kein Countdown angezeigt, bei `false` wird er entsprechend angezeigt.
- ▶ **GRUB_CMDLINE_LINUX=**
Hiermit werden Standardoptionen für jede `linux`-Zeile gesetzt.

Die Variablen werden erst nach einem erneuten Aufruf von `update-grub` gültig.

2.2 Bootloader Recovery

Es passiert selten, aber wenn Sie Ihr System aufgrund einer Fehlkonfiguration des Bootloaders nicht mehr starten können, sollten Sie den Bootloader reparieren. Dazu können Sie den Rechner von einer beliebigen Live-CD¹ oder DVD oder von einem USB-Stick neu starten.



Der einfachste Weg, eine Reparatur durchzuführen, ist, die Live-CD des Systems zu verwenden, mit der Sie den Rechner installiert haben. Beachten Sie jedoch, dass Sie in jedem Fall bei Benutzung einer anderen Rettungs-CD dieselbe Architektur verwenden, die auch Ihr installiertes System aufweist.

Nach dem Start des Rettungssystems wird die Festplatte Ihres defekten Systems eingebunden. Das bedeutet, dass Sie alle Partitionen *mounten*. Im Regelfall werden die Partitionen unter */mnt* eingebunden. Sie können natürlich auch eigene Verzeichnisse verwenden, wenn Sie dabei keines der vom Live-System benutzten Verzeichnisse einsetzen.

Das Kommando `fdisk -l` zeigt Ihnen alle gefundenen Festplatten an. Falls Software-RAIDs oder LVM benutzt werden, müssen diese vor der Benutzung aktiviert werden. Wie das geht, erklären wir in Kapitel 3, »Festplatten und andere Devices«.

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	2048	499711	248832	83	Linux
/dev/sda2		501758	20969471	10233857	5	Extended
/dev/sda5		501760	20969471	10233856	8e	Linux LVM

Listing 2.6 Ausgabe von »`fdisk -l`« auf einem Testsystem

In Listing 2.6 finden Sie den seltenen Fall eines Systems ohne eigene *Swap*-Partition. Vermutlich findet sich in der ersten Partition das *boot*-Verzeichnis, was wir durch *Mounten* verifizieren können (siehe Listing 2.7):

```
Rescue:~# mount /dev/sda1 /mnt
Rescue:~# ls /mnt
System.map-3.16.0-4-amd64  grub                                vmlinuz-3.16.0-4-amd64
config-3.16.0-4-amd64    initrd.img-3.16.0-4-amd64
```

Listing 2.7 Mounten des vermeintlichen »boot«-Filesystems

Die fünfte Partition wird vom *Logical Volume Manager* (LVM) verwaltet.

Mit dem Kommando `lvs` (siehe Listing 2.8) können wir uns die gefundenen *Logical Volumes* anzeigen lassen. Der Parameter `-o +lv_path` sorgt dafür, dass uns auch gleich der Pfad für das *Mounten* gezeigt wird:

¹ Zum Beispiel die »System Rescue CD«, <https://www.system-rescue.org/>

```
Rescue:~# lvs -o +lv_path
  LV      VG      Attr      LSize   Pool Origin Data%  Move Log Copy%  Convert \
Path
  root    debian -wi-ao--  9.31g                               \
/dev/debian/root
  swap_1  debian -wi-ao--  460.00m                               \
/dev/debian/swap_1
```

Listing 2.8 Gefundene Partition im LVM

An dieser Stelle haben wir alle Informationen zusammen, um die Dateisysteme benutzen zu können (siehe Listing 2.9):

```
Rescue:~ # mount /dev/debian/root /mnt
Rescue:~ # mount /dev/sda1 /mnt/boot/
```

Listing 2.9 Mounten der Dateisysteme

Um das Linux-System komplett zu machen, müssen wir die dynamischen Pseudo-Dateisysteme (*/dev*, */proc* und */sys*) aus der Live-CD in die Verzeichnisse unterhalb von */mnt* einbinden. Das funktioniert über *Bind-Mounts*. Wenn das nicht passieren würde, erhalten wir nach einem Wechsel der root-Umgebung mittels *chroot* (»change root environment«) keine Informationen über verbundene Geräte und Kernelparameter (siehe Listing 2.10):

```
Rescue:~ # mount --bind /dev /mnt/dev
Rescue:~ # mount --bind /proc /mnt/proc
Rescue:~ # mount --bind /sys /mnt/sys
```

Listing 2.10 Bind-Mount der Pseudodateisysteme

Damit sind jetzt alle Vorarbeiten abgeschlossen, um via *chroot* auf das System zu wechseln und den Bootloader zu reparieren (siehe Listing 2.11):

```
Rescue:~ # chroot /mnt
Rescue:/ # grub-install
Rescue:/ # exit
```

Listing 2.11 Neuinstallation des Bootloaders

Sobald Sie fertig sind, müssen alle Dateisysteme ausgehängt werden. Anschließend müssen Sie das System neu starten.

2.3 Der Kernel und die initrd

Beim Laden des Kernels gibt es ein klassisches Henne-Ei-Problem: Der Kernel probiert nämlich zunächst, alle notwendigen Module zu laden, die für den Zugriff auf die Hardware not-

wendig sind. Das sind insbesondere die Treiber zum Ansprechen der Festplatte und des Dateisystems. Die dafür notwendigen Module liegen aber auf dem noch nicht lesbaren Dateisystem. Um dieses Dilemma zu lösen, lädt der Bootloader nicht nur den Kernel direkt in den Speicher, sondern auch die *Initial Ramdisk* (*initrd*). Die *initrd* besteht aus einem komprimierten *cpio*-Archiv und enthält ein absolut minimales Linux mit allen für den Start notwendigen Modulen. Der Kernel benutzt die *initrd* als *root*-Dateisystem. Sobald alle nötigen Treiber geladen sind, bindet der Kernel das eigentliche *root*-Dateisystem ein und startet den *systemd*-Prozess.

2.3.1 *initrd* erstellen und modifizieren

Bei der Installation eines Systems wird auch eine *Initial Ramdisk* (*initrd*) erstellt, die Treiber enthält, die für den Start des Rechners benötigt werden, bevor die Dateisysteme verfügbar sind. Diese Ramdisk wird bei jedem Kernelupdate neu erstellt und mit neuen Versionen der Treiber versehen. Wenn Sie allerdings Hardware benutzen, die Treiber benötigt, die nicht im Kernel vorhanden sind, wie beispielsweise besondere *RAID*-Controller oder Netzwerkkarten, so müssen Sie – wenn Sie Ihr System von den Geräten aus starten wollen – selbst Hand anlegen, falls das nicht die Installationsroutine des Herstellers für Sie übernimmt. Die verschiedenen Distributionen nutzen unterschiedliche Tools für die Erstellung. In den folgenden Abschnitten finden Sie die Beschreibungen für die im Buch unterstützten Distributionen, gefolgt von einem Abschnitt über die komplett manuelle Erstellung der Initial Ramdisk.

Debian und Ubuntu

Debian und Ubuntu benutzen `mkinitramfs` und `update-initramfs`. Wenn Sie nicht besondere Gründe haben, sollten Sie immer `update-initramfs` verwenden, da dieses Kommando unter anderem auch `mkinitramfs` auf Basis der bereits bestehenden Konfiguration aufruft.

Die Erstellung der *initrd* wird über die Konfigurationsdatei `/etc/initramfs-tools/initramfs.conf` und weitere Dateien innerhalb des Verzeichnisses `/etc/initramfs-tools` gesteuert. Aufgrund der vielen Kommentare in den Dateien werden Sie schnell zum Ziel kommen.

Einen besonderen Blick verdient die wichtigste Variable, `MODULES`. Sie kann verschiedene Werte annehmen, wie folgende Auflistung zeigt:

- ▶ **most**
Das ist die Standardeinstellung bei Ubuntu und Debian. Damit werden fast alle Dateisystem- und Hardwaretreiber übernommen. Die daraus resultierende sehr große Initial Ramdisk kann dafür aber auch fast jedes System starten.
- ▶ **dep**
Das laufende System wird analysiert, um festzustellen, welche Module wichtig sind. Diese Einstellung verkleinert die Initial Ramdisk auf ein Minimum.

► **netboot**

Wie der Name es beschreibt, werden mit dieser Einstellung nur Treiber verwendet, die für das Starten über das Netzwerk nötig sind.

► **list**

Ausschließlich Module aus `/etc/initramfs-tools/modules` werden zum Bau der Initial Ramdisk verwendet. Dies erlaubt die größtmögliche Kontrolle.

Auch ohne weitere Konfiguration werden die Module aus `/etc/initramfs-tools/modules` bei den Parametern `most`, `dep` und `netboot` zur Initial Ramdisk hinzugefügt.



Die Konfigurationen in den Dateien unterhalb von `/etc/initramfs-tools/conf.d` können die Werte aus `/etc/initramfs-tools/initramfs.conf` überschreiben.



Um eine neue Initial Ramdisk zu erstellen bzw. die bestehende aktualisieren zu lassen, können Sie mit `update-initramfs` den Neubau starten. Die unten stehenden Parameter helfen bei der Erstellung:

► **update-initramfs -u**

Hiermit werden alle vorhandenen Initial Ramdisks aktualisiert.

► **update-initramfs -k KERNEL**

Dieser Parameter wird benötigt, wenn nur die Initial Ramdisks einer bestimmten Kernelversion aktualisiert werden sollen.

► **update-initramfs -c**

Dieser Parameter erstellt komplett neue Initial Ramdisks.

Der Name der Initial Ramdisk ergibt sich aus dem Namen des Kernels. Eine vorhandene Ramdisk wird somit bei jedem Aufruf von `update-initramfs` überschrieben.

Wenn Sie dieses Verhalten nicht wünschen, sollten Sie den Parameter `backup_initramfs=yes` in der Datei `/etc/initramfs-tools/update-initramfs.conf` setzen oder manuelle Backups erstellen (siehe Listing 2.12):

```
root@debian:~# update-initramfs -v -k 3.16.0-4-amd64 -c
update-initramfs: Generating /boot/initrd.img-3.16.0-4-amd64
Copying module directory kernel/drivers/hid
(excluding hid-*.ko hid-a4tech.ko hid-cypress.ko hid-dr.ko hid-elecom.ko \
hid-gyration.ko hid-icade.ko hid-kensington.ko hid-kye.ko hid-lcpower.ko \
hid-magicmouse.ko hid-multitouch.ko hid-ntrig.ko hid-petalynx.ko \
hid-picolcd.ko hid-pl.ko hid-ps3remote.ko hid-quanta.ko hid-roccat-ko*.ko \
hid-roccat-pyra.ko hid-saitek.ko hid-sensor-hub.ko hid-sony.ko \
hid-speedlink.ko hid-tivo.ko hid-twinhan.ko hid-uclogic.ko hid-wacom.ko \
hid-waltop.ko hid-wiimote.ko hid-zydacron.ko)
Adding module /lib/modules/3.16.0-4-amd64/kernel/drivers/hid/hid.ko
[...]
Adding library /lib/x86_64-linux-gnu/librt.so.1
```

```
Adding module /lib/modules/3.16.0-4-amd64/kernel/drivers/md/dm-mod.ko
/usr/share/initramfs-tools/scripts/local-premount/ORDER ignored: not executable
/usr/share/initramfs-tools/scripts/init-top/ORDER ignored: not executable
/usr/share/initramfs-tools/scripts/init-bottom/ORDER ignored: not executable
Building cpio /boot/initrd.img-3.16.0-4-amd64.new initramfs
```

Listing 2.12 Neuerstellen einer »initrd«



Wenn der Name der Initial Ramdisk bereits existierte, ist nichts weiter zu tun. Sollten Sie aber einen neuen Namen verwenden, muss im Bootloader der entsprechende Name eingetragen werden, sonst können Sie das System nicht mehr starten.

CentOS und openSUSE

Anders als bei Ubuntu und Debian nutzen CentOS und openSUSE das Skript `mkinitrd`, um eine Initial Ramdisk zu erstellen. Das Skript ermittelt die Treiber, die aufgenommen werden müssen, und nutzt die Informationen aus `/etc/sysconfig/kernel`, in der eine Liste von Modulen zu finden ist, die zusätzlich hinzugefügt werden sollen (siehe Listing 2.13):

```
Creating initrd: /boot/initrd-4.1.27-27-default
Executing: /usr/bin/dracut --logfile /var/log/YaST2/mkinitrd.log \
--force /boot/initrd-4.1.27-27-default 4.1.27-27-default
...
*** Including module: bash ***
*** Including module: warpclock ***
*** Including module: i18n ***
*** Including module: ifcfg ***
*** Including module: btrfs ***
*** Including module: kernel-modules ***
Omitting driver i2o_scsi
*** Including module: resume ***
*** Including module: rootfs-block ***
*** Including module: terminfo ***
*** Including module: udev-rules ***
Skipping udev rule: 91-permissions.rules
Skipping udev rule: 80-drivers-modprobe.rules
*** Including module: haveged ***
*** Including module: systemd ***
*** Including module: usrmount ***
*** Including module: base ***
*** Including module: fs-lib ***
*** Including module: shutdown ***
*** Including module: suse ***
*** Including modules done ***
*** Installing kernel module dependencies and firmware ***
```

```
*** Installing kernel module dependencies and firmware done ***
*** Resolving executable dependencies ***
*** Resolving executable dependencies done***
*** Hardlinking files ***
*** Hardlinking files done ***
*** Stripping files ***
*** Stripping files done ***
*** Generating early-microcode cpio image ***
*** Store current command line parameters ***
Stored kernel commandline:
  resume=UUID=54c1a2e0-c4d6-4850-b639-7a5af8ef4339
root=UUID=0f4f79aa-7544-44b0-a8b7-d1f1947cd24f \
rootflags=rw,relatime,space_cache,subvolid=257,subvol=/@ rootfstype=btrfs
*** Creating image file ***
*** Creating image file done ***
Some kernel modules could not be included
This is not necessarily an error:
*** Including module: udev-rules ***
Skipping udev rule: 91-permissions.rules
Skipping udev rule: 80-drivers-modprobe.rules
*** Including module: haveged ***
*** Including module: systemd ***
*** Including module: usrmount ***
*** Including module: base ***
*** Including module: fs-lib ***
*** Including module: shutdown ***
*** Including module: suse ***
*** Including modules done ***
*** Installing kernel module dependencies and firmware ***
*** Installing kernel module dependencies and firmware done ***
*** Resolving executable dependencies ***
*** Resolving executable dependencies done***
*** Hardlinking files ***
*** Hardlinking files done ***
*** Stripping files ***
*** Stripping files done ***
*** Generating early-microcode cpio image ***
*** Store current command line parameters ***
Stored kernel commandline:
  resume=UUID=54c1a2e0-c4d6-4850-b639-7a5af8ef4339
root=UUID=0f4f79aa-7544-44b0-a8b7-d1f1947cd24f \
rootflags=rw,relatime,space_cache,subvolid=257,subvol=/@ rootfstype=btrfs
*** Creating image file ***
```

```
*** Creating image file done ***  
Some kernel modules could not be included  
This is not necessarily an error:
```

Listing 2.13 Beispiel für »mkinitrd« unter openSUSE

Die Installation des Systems setzt automatisch die Variable `INITRD_MODULES`. Wenn diese Liste um eigene Einträge ergänzt wird, muss anschließend `mkinitrd` aufgerufen werden.

Analog zu `update-initramfs` bei Ubuntu und Debian bietet auch `mkinitrd` einige Optionen an, die Ihnen helfen, die Initial Ramdisk anzupassen:

- ▶ **-k KERNEL**
Angabe des Kernels, für den die Initial Ramdisk gebaut werden soll. Ohne Angabe des Parameters wird `vmlinuz` benutzt.
- ▶ **-i INITRD**
setzt den Namen der Initial Ramdisk. Ohne diese Angabe wird `/boot/initrd` genommen.
- ▶ **-m MODULES**
nimmt eine Liste von Modulen auf der Kommandozeile, ansonsten wird der Inhalt der Variablen `INITRD_MODULES` aus `/etc/sysconfig/kernel` ausgelesen.
- ▶ **-f FEATURES**
setzt Funktionalitäten für den Kernel. Abhängig davon werden weitere Module und Skripte eingebunden. Als Beispiel seien hier *Software-RAID* (Parameter `dm`) und *Logical Volume Manager* (Parameter `lvm2`) genannt.

In Listing 2.14 sehen Sie einen Beispielaufruf von `mkinitrd`:

```
opensuse:~ # mkinitrd -k 4.1.27-27-default -i initrdtest -m ext4 \  
-f "lvm2 dm block"
```

Listing 2.14 Beispielaufruf von »mkinitrd«

2.3.2 initrd manuell modifizieren

Zusätzlich zu den vorgestellten Methoden, die zugegebenermaßen relativ beschränkt sind, lässt sich die Initial Ramdisk (`initrd`) auch manuell verändern.

Als Basis für Ihre Arbeiten nehmen Sie sich bitte eine vorhandene `initrd` und packen diese aus. Listing 2.15 zeigt Ihnen, dass es sich bei der `initrd` um ein minimales `root`-Filesystem handelt:

```
root@debian:~# mkdir /var/tmp/initrd  
root@debian:~# cd /var/tmp/initrd/  
root@debian:/var/tmp/initrd# gzip -dc /boot/initrd.img-3.16.0-4-amd64 \  
| cpio --extract --make-directories
```



```

90084 blocks
root@debian:/var/tmp/initrd# ls -l
total 40
drwxr-xr-x 2 root root 4096 Aug  4 15:31 bin
drwxr-xr-x 3 root root 4096 Aug  4 15:31 conf
drwxr-xr-x 5 root root 4096 Aug  4 15:31 etc
-rwxr-xr-x 1 root root 7137 Aug  4 15:31 init
drwxr-xr-x 7 root root 4096 Aug  4 15:31 lib
drwxr-xr-x 2 root root 4096 Aug  4 15:31 lib64
drwxr-xr-x 2 root root 4096 Aug  4 15:31 run
drwxr-xr-x 2 root root 4096 Aug  4 15:31 sbin
drwxr-xr-x 5 root root 4096 Aug  4 15:31 scripts

```

Listing 2.15 »initrd« entpacken

In dem resultierenden Verzeichnis `/var/tmp/initrd` können Sie nun Ihre Änderungen einpflegen und danach alles wieder einpacken (siehe Listing 2.16):

```

root@debian:/var/tmp/initrd# find . \
| cpio --create --format=newc \
| gzip > /boot/initrd.adminbuch
90084 blocks

```

Listing 2.16 »initrd« einpacken

In der Datei `/boot/initrd.adminbuch` findet sich nun die `initrd`, die alle Ihre Änderungen enthält.

2.4 systemd

Nach dem Bootvorgang, in dem der Kernel das `root`-Filesystem eingebunden und alle notwendigen Module geladen hat, übernimmt der `systemd`-Daemon den weiteren Ablauf. Der klassische `init`-Prozess folgt dem in System V² vorgestellten Verfahren und wird nach diesem auch `SysVinit` genannt. Er ist verantwortlich für das Starten der Dienste in der richtigen Reihenfolge, für das Folgen und auch für den Wechsel von Runleveln sowie für das Stoppen von Prozessen. Dieses Verfahren ist sehr robust, aber leider auch sehr statisch.

`systemd` ist der Nachfolger, den mittlerweile alle Distributionen verwenden. Mit `systemd` gibt es einen Übergang vom statischen Starten von Skripten zum eventbasierten Starten. So können Bedingungen definiert werden, die erfüllt sein müssen, um Dienste starten zu können (beispielsweise wird der Webserver erst dann gestartet, wenn das Netzwerk verfügbar ist, oder ein Virens scanner erst dann, wenn ein USB-Stick eingesteckt wird). Der Start von

² https://de.wikipedia.org/wiki/System_V

Diensten mit `systemd` ist im Unterschied zu `SysVinit` hoch parallelisierbar. Als besonderes Feature ist `systemd` auch in der Lage, abgestürzte Dienste neu zu starten. Es gibt kaum ein Thema in den letzten Jahren, das in der Linux-Community so kontrovers diskutiert wurde, wie die Einführung von `systemd`.

`systemd` schickt sich an, den kompletten altbekannten und bewährten Bootvorgang auf den Kopf zu stellen. Den einen gehen die Änderungen zu weit, die anderen feiern mit `systemd` die Ankunft im neuen Jahrtausend. Tatsache ist, dass mit `systemd` Start-Skripte – genauer gesagt Startkonfigurationen – parallel ausgeführt werden können und nicht wie früher linear. Dazu kommt, dass `systemd` Programme voneinander kapselt und in eigenen *Control Groups* und *Namespaces* startet und so sicherstellt, dass beim Beenden eines Dienstes auch alle Prozesse im gleichen Namespace mit beendet werden und dass es keine verwaisten Prozesse gibt.

Weitergehende Änderungen sind, dass mit *journald* ein eigenes Logging-Framework mitgeliefert wird, das es erlaubt, fälschungssichere Logs zu führen. Dadurch soll das altbekannte *syslog* abgelöst werden. *timers* in `systemd` sind in der Lage, klassische Cron- und Anacron-Jobs abzulösen, *systemd-mounts* könnten die *fstab* überflüssig machen. Die beiden Hauptkritikpunkte der `systemd`-Gegner sind, dass `systemd` von der UNIX-Philosophie »one task, one tool« abweicht und dass das `systemd`-Team bei der Weiterentwicklung zum Teil fragwürdige Entscheidungen trifft.

2.4.1 Begriffe

`systemd` wird mit *Units* verwaltet. *Units* kapseln verschiedene Aspekte eines Systems und können untereinander in Beziehung gesetzt werden. Die Definitionen der *Units* sind einfache Textdateien, die ein wenig an ini-Dateien aus Windows erinnern. Die einzelnen *Unit*-Typen sind die folgenden:


- ▶ **Service Units**
werden benutzt, um Dienste und Prozesse zu starten.
- ▶ **Socket Units**
kapseln IPC- oder Netzwerk-Sockets, die vor allem gebraucht werden, um Socket-basierend Dienste zu aktivieren.
- ▶ **Target Units**
können zur Gruppierung von Diensten oder zur Erstellung von Synchronisationspunkten benutzt werden (hiermit lassen sich Runlevel wie im `SysVinit` emulieren).
- ▶ **Device Units**
sind die Verbindung zu Kernel-Devices und können ebenfalls benutzt werden, um Device-basierende Dienste zu steuern.
- ▶ **Mount Units**
kontrollieren Mountpunkte im System.

- ▶ **Automount Units**
werden für zugriffsbasiertes Einbinden von Dateisystemen benutzt und dienen insbesondere auch der Parallelisierung im Bootprozess.
- ▶ **Snapshot Units**
können den Status einer Anzahl von systemd-Units aufzeichnen und diesen Status durch Aktivierung auch wiederherstellen.
- ▶ **Timer Units**
bieten die Möglichkeit, eine zeitbasierte Steuerung anderer Units vorzunehmen.
- ▶ **Swap Units**
verwalten – analog zu Mount Units – Swap-Speicherplatz.
- ▶ **Path Units**
aktivieren bei einer Veränderung von Dateisystemobjekten andere Dienste.
- ▶ **Slice Units**
gruppieren Units in hierarchischer Form, die Systemprozesse – beispielsweise Service oder Scope Units – verwalten.
- ▶ **Scope Units**
gleichem Service Units, verwalten aber Fremdprozesse, anstatt sie nur zu starten.

Wie Sie allein an den verschiedenartigen Units feststellen können, kann man in systemd vielfältige Aspekte eines Systems beeinflussen. Im Folgenden gehen wir auf System Units ein – das sind die Units, mit denen Sie am häufigsten in Kontakt kommen werden.

2.4.2 Kontrollieren von Diensten

Das Hauptkommando, mit dem Sie systemd kontrollieren können, heißt `systemctl`. Dieser Befehl wird auch benutzt, um Dienste zu verwalten. Analog zu den früheren `init`-Skripten gibt es die Kommandos `start`, `stop`, `reload`, `restart` und `status`.

`systemctl` macht Gebrauch von Farben im Terminal. Stellen Sie daher bitte sicher, dass Ihr Terminal auch Farben darstellen kann. 

Service Units in systemd enden auf `.service`. Diese Endung muss aber nicht explizit angegeben werden. In Listing 2.17 sehen Sie, dass weder das Stopp- noch das Start-Subkommando sehr Gesprächig ist, daher sollte das Ergebnis mit einer Statusabfrage überprüft werden:

```
# systemctl stop sshd
# systemctl status sshd
* ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: inactive (dead) since Sat 2021-01-30 11:30:01 UTC; 6s ago
    Docs: man:sshd(8)
          man:sshd_config(5)
```

```
Process: 626 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
Process: 640 ExecStart=/usr/sbin/sshd -D $SSHD_OPTS (code=exited, status=0/SUCCESS)
Main PID: 640 (code=exited, status=0/SUCCESS)

Jan 30 09:52:18 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Jan 30 09:52:18 ubuntu sshd[640]: Server listening on 0.0.0.0 port 22.
Jan 30 09:52:18 ubuntu sshd[640]: Server listening on :: port 22.
Jan 30 09:52:18 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
Jan 30 09:53:02 ubuntu sshd[796]: Accepted publickey for root from 10.0.2.2 port \
    44366 ssh2: ED25519 SHA256:Z5Y3PGikMomjoHevFwBo6XFRUsZ6Fe/Xc9RmYNLSxXk
Jan 30 09:53:02 ubuntu sshd[796]: pam_unix(sshd:session): session opened for user \
    root by (uid=0)
Jan 30 11:30:01 ubuntu systemd[1]: Stopping OpenBSD Secure Shell server...
Jan 30 11:30:01 ubuntu sshd[640]: Received signal 15; terminating.
Jan 30 11:30:01 ubuntu systemd[1]: ssh.service: Succeeded.
Jan 30 11:30:01 ubuntu systemd[1]: Stopped OpenBSD Secure Shell server.

# systemctl start sshd
# systemctl status sshd
* ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2021-01-30 11:30:13 UTC; 3s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 1700 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
  Main PID: 1711 (sshd)
    Tasks: 1 (limit: 1074)
   Memory: 2.9M
   CGroup: /system.slice/ssh.service
           └─1711 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
```

```
Jan 30 11:30:13 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
Jan 30 11:30:13 ubuntu sshd[1711]: Server listening on 0.0.0.0 port 22.
Jan 30 11:30:13 ubuntu sshd[1711]: Server listening on :: port 22.
Jan 30 11:30:13 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
```

Listing 2.17 Stoppen des SSH-Servers

Gerade die Statusausgaben sind auf den ersten Blick sehr verwirrend:

- ▶ In der ersten Zeile finden Sie den Namen des Dienstes und die Beschreibung.
- ▶ Die Zeile, die mit Loaded: beginnt, zeigt Ihnen, ob die Unit-Datei geladen ist, und den Speicherort. Mit enabled ist gemeint, dass die Unit standardmäßig (z. B. beim Start des Systems) ausgeführt wird und wie die Einstellung des Distributors (CentOS, Debian, open-

SUSE oder Ubuntu) ist. Mehr Informationen dazu finden Sie in Abschnitt 2.4.3, »Aktivieren und Deaktivieren von Diensten«.

- ▶ **Active:** kennzeichnet den aktuellen Status und gibt an, seit wann dieser Status besteht.
- ▶ **Docs:** verweist auf Dokumentationen zum Service. Hier sind es Manpages, üblich ist aber auch der Hinweis auf eine URL.
- ▶ **Process:** zeigt Ihnen, wie der Dienst gestartet wurde, und gibt den letzten Status an.
- ▶ **Main PID:** enthält die Hauptprozess-ID.
- ▶ **CGroup:** stellt die Control Group dar, in der der Dienst gestartet wurde.
- ▶ Zum Schluss folgt ein Auszug der letzten Log-Ausgaben. Wie Sie mehr Log-Ausgaben sehen können, erfahren Sie in Abschnitt 2.4.8, »Logs mit journald«.

Die Subkommandos `restart`, um den Dienst neu zu starten, und `reload`, um die Konfiguration – in diesem Fall `/etc/ssh/sshd_config` – neu einzulesen, vervollständigen die Basiskommandos.

Folgende Befehle wurden in diesem Abschnitt behandelt:

- ▶ `systemctl start <SERVICE>`
- ▶ `systemctl stop <SERVICE>`
- ▶ `systemctl status <SERVICE>`
- ▶ `systemctl restart <SERVICE>`
- ▶ `systemctl reload <SERVICE>`

2.4.3 Aktivieren und Deaktivieren von Diensten

Units werden mit den Subkommandos `enable` und `disable` aktiviert und deaktiviert:

```
# systemctl disable sshd
Removed symlink /etc/systemd/system/multi-user.target.wants/sshd.service.

# systemctl enable sshd
Created symlink from /etc/systemd/system/multi-user.target.wants/sshd.service \
to /usr/lib/systemd/system/sshd.service.
```

Listing 2.18 Aktivieren und Deaktivieren des SSH-Servers

Die Subkommandos lesen die Servicedefinition und schauen, für welches Target der Dienst aktiviert werden soll. In Listing 2.18 sehen Sie, dass es das `multi-user.target` für den Dienst `sshd` ist. Beim Aktivieren wird nun ein Link in dem Verzeichnis des Targets erstellt und beim Deaktivieren wieder gelöscht.

Folgende Befehle wurden in diesem Abschnitt behandelt:

- ▶ `systemctl enable <SERVICE>`
- ▶ `systemctl disable <SERVICE>`

2.4.4 Erstellen und Aktivieren eigener Service Units

Wie Sie bereits in Listing 2.17 im Abschnitt 2.4.2 gesehen haben, ist die Konfigurationsdatei `/usr/lib/systemd/system/sshd.service` die Datei, in der die Angaben des SSH-Servers gespeichert werden (siehe Listing 2.19). Viele Ausgaben des Statuskommandos werden durch Einträge im Servicefile angezeigt, beispielsweise die `Description` oder `Documentation`:

```
# systemctl cat sshd
# /usr/lib/systemd/system/sshd.service
[Unit]
Description=OpenSSH server daemon
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.service
Wants=sshd-keygen.service

[Service]
EnvironmentFile=/etc/sysconfig/ssh
ExecStart=/usr/sbin/sshd -D $OPTIONS
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartSec=42s

[Install]
WantedBy=multi-user.target
```

Listing 2.19 Inhalt von »sshd.service«

Die Ausgabe von `systemctl show sshd` zeigt Ihnen neben den Einträgen aus dem Servicefile auch noch die ganzen Standardoptionen und Statusinformationen an. Die 143 Zeilen Information ersparen wir Ihnen an dieser Stelle, da Sie diese jederzeit selbst abrufen können. Die einzelnen Parameter der Konfiguration aus Listing 2.19 haben die folgende Bedeutung:

- ▶ **Description**
enthält eine lesbare Beschreibung des Dienstes.
- ▶ **Documentation**
verweist auf weiterführende Informationen.

- ▶ **After**
wird benutzt, um Abhängigkeiten zu definieren. In diesem Fall soll der Dienst nach den angegebenen anderen Diensten gestartet werden (analog dazu gibt es `Before`).
- ▶ **Wants**
erfordert, dass die angegebenen Dienste vor dem Start erfolgreich gelaufen sind (abgemilderte Form von `Require`).
- ▶ **EnvironmentFile**
gibt eine Datei mit Umgebungsvariablen an, die zur Verfügung stehen sollen.
- ▶ **ExecStart**
enthält das Kommando, das zum Start benutzt wird (analog dazu gibt es `ExecStop`).
- ▶ **ExecReload**
Mit diesem Kommando werden die Konfigurationsdateien neu eingelesen.
- ▶ **KillMode**
zeigt, mit welchem Verfahren der Prozess getötet werden kann.
- ▶ **Restart**
definiert die Option für den automatischen Neustart des Dienstes.
- ▶ **RestartSec**
enthält die Zeit, nach der neu gestartet werden soll.
- ▶ **WantedBy**
beschreibt das Target (oder den Service), durch das der Dienst automatisch gestartet werden soll.

Zusätzlich zu den Optionen, die Sie in der Definition des SSH-Servers sehen, gibt es noch die folgenden Optionen, denen Sie häufiger begegnen werden:

- ▶ **Before**
wird analog zu `After` benutzt, um Abhängigkeiten zu definieren. In diesem Fall soll der Dienst vor den angegebenen anderen Diensten gestartet werden.
- ▶ **Require**
erfordert, dass die angegebenen Dienste vor dem Start erfolgreich gelaufen sind. Wenn die Dienste beendet werden, soll unser Dienst ebenfalls gestoppt werden.
- ▶ **ExecStop**
enthält das Kommando, das zum Stoppen benutzt wird.
- ▶ **Conflicts**
beendet die angegebenen Dienste, wenn der jetzt konfigurierte Dienst gestartet wird.
- ▶ **OnFailure**
enthält eine Liste an Units, die gestartet werden, wenn sich dieser Service fehlerhaft beendet.

► Type

ist für Services entweder `simple` oder `forking`, wobei das Erste für einen Prozess steht, der ständig läuft, und das Zweite für einen Prozess, der einen Kindprozess abspaltet und sich danach beendet. (`oneshot` ist ein Service, der nur läuft und sich danach selbst beendet. Dieser Typ wird manchmal als Ziel für `OnFailure` benutzt.)



Eigene Servicedateien sollten Sie im dafür vorgesehenen Verzeichnis `/etc/systemd/system` anlegen und nach dem Anlegen mittels `systemctl daemon-reload` aktivieren.

Weitergehende Informationen finden Sie in der Manpage `systemd.unit`.

Folgende Befehle wurden im aktuellen Abschnitt behandelt:

- `systemctl show <SERVICE>`
- `systemctl cat <SERVICE>`

2.4.5 Target Units

Die folgenden Targets finden sich auf einem Desktop-System:

```
# systemctl list-units "*.target"
```

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
basic.target	loaded	active	active	Basic System
bluetooth.target	loaded	active	active	Bluetooth
cryptsetup.target	loaded	active	active	Encrypted Volumes
getty.target	loaded	active	active	Login Prompts
graphical.target	loaded	active	active	Graphical Interface
local-fs-pre.target	loaded	active	active	Local File Systems (Pre)
local-fs.target	loaded	active	active	Local File Systems
multi-user.target	loaded	active	active	Multi-User System
network-online.target	loaded	active	active	Network is Online
network.target	loaded	active	active	Network
nfs-client.target	loaded	active	active	NFS client services
paths.target	loaded	active	active	Paths
remote-fs-pre.target	loaded	active	active	Remote File Systems (Pre)
remote-fs.target	loaded	active	active	Remote File Systems
slices.target	loaded	active	active	Slices
sockets.target	loaded	active	active	Sockets
sound.target	loaded	active	active	Sound Card
swap.target	loaded	active	active	Swap
sysinit.target	loaded	active	active	System Initialization
timers.target	loaded	active	active	Timers

LOAD = Reflects whether the unit definition was properly loaded.
 ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
 SUB = The low-level unit activation state, values depend on unit type.

20 loaded units listed. Pass --all to see loaded but inactive units, too.
 To show all installed unit files use 'systemctl list-unit-files'.

Listing 2.20 Targets auf einem Desktop-System

Mittels `systemctl list-dependencies multi-user.target` können Sie sich anzeigen lassen, von welchen Diensten das Multi-User-Target abhängt.

Folgender Befehl wurde in diesem Abschnitt behandelt:

▶ `systemctl list-units '*.targets'`

2.4.6 »systemd«- und Servicekonfigurationen

Ein installiertes System kommt mit einer großen Anzahl an Diensten daher. Um da nicht den Überblick zu verlieren, bietet `systemd` einige Kommandos, mit denen Sie das laufende System abfragen können. Listing 2.21 zeigt Ihnen einen Auszug der 228 bekannten Units auf einem minimal installierten CentOS-System:

UNIT FILE	STATE
[...]	
crond.service	enabled
fstrim.service	static
kdump.service	enabled
NetworkManager.service	enabled
postfix.service	enabled
sshd.service	enabled
ctrl-alt-del.target	disabled
graphical.target	static
hibernate.target	static
hybrid-sleep.target	static
network-online.target	static
runlevel0.target	disabled
runlevel1.target	disabled
runlevel2.target	static
runlevel3.target	static
runlevel4.target	static
runlevel5.target	static
runlevel6.target	disabled

[...]
228 unit files listed.

Listing 2.21 Auszug der bekannten Units eines Systems

Von diesen 228 bekannten Units wurden aber nur 96 geladen:

```
# systemctl list-units
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
[...]
-.mount                             loaded active mounted /
boot.mount                          loaded active mounted /boot
dbus.service                        loaded active running D-Bus System Message Bus
getty@tty1.service                  loaded active running Getty on tty1
NetworkManager.service             loaded active running Network Manager
getty.target                        loaded active active Login Prompts
local-fs.target                    loaded active active Local File Systems
multi-user.target                   loaded active active Multi-User System
network-online.target               loaded active active Network is Online
network.target                      loaded active active Network
systemd-tmpfiles-clean.timer        loaded active waiting Daily Cleanup of \
                                     Temporary Directories
[...]
LOAD    = Reflects whether the unit definition was properly loaded.
ACTIVE  = The high-level unit activation state, i.e. generalization of SUB.
SUB     = The low-level unit activation state, values depend on unit type.
```

96 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.

Listing 2.22 Auszug der geladenen Unit-Dateien

Folgender Befehl wurde in diesem Abschnitt behandelt:

▶ systemctl list-units

2.4.7 Anzeige von Dienstabhängigkeiten

Wenn man sich die Voraussetzungen anschauen möchte, die erforderlich sind, um eine Unit starten zu können, kann man das Subkommando list-dependencies benutzen:

```
# systemctl list-dependencies sshd.service
sshd.service
* |-sshd-keygen.service
```

```

* |-system.slice
* `--basic.target
*   |-microcode.service
*   |-rhel-autorelabel-mark.service
*   |-rhel-autorelabel.service
*   |-rhel-configure.service
*   |-rhel-dmesg.service
*   |-rhel-loadmodules.service
*   |-paths.target
*   |-slices.target
*   | |--.slice
*   | `--system.slice
*   |-sockets.target
*   | |-dbus.socket
*   | |-dm-event.socket
*   | |-systemd-initctl.socket
*   | |-systemd-journald.socket
*   | |-systemd-shutdown.socket
*   | |-systemd-udev-control.socket
*   | `--systemd-udev-kernel.socket
*   |-sysinit.target
*   | |-dev-hugepages.mount
*   | |-dev-mqueue.mount
*   | |-kmod-static-nodes.service
*   | |-lvm2-lvmetad.socket
*   | |-lvm2-lvmpolld.socket
*   | |-lvm2-monitor.service
*   | |-plymouth-read-write.service
*   | |-plymouth-start.service
*   | |-proc-sys-fs-binfmt_misc.automount
*   | |-sys-fs-fuse-connections.mount
*   | |-sys-kernel-config.mount
*   | |-sys-kernel-debug.mount
*   | |-systemd-ask-password-console.path
*   | |-systemd-binfmt.service
*   | |-systemd-firstboot.service
*   | |-systemd-hwdb-update.service
*   | |-systemd-journal-catalog-update.service
*   | |-systemd-journal-flush.service
*   | |-systemd-journald.service
*   | |-systemd-machine-id-commit.service
*   | |-systemd-modules-load.service
*   | |-systemd-random-seed.service

```

```
* | |-systemd-sysctl.service
* | |-systemd-tmpfiles-setup-dev.service
* | |-systemd-tmpfiles-setup.service
* | |-systemd-udev-trigger.service
* | |-systemd-udevd.service
* | |-systemd-update-done.service
* | |-systemd-update-utmp.service
* | |-systemd-vconsole-setup.service
* | |-cryptsetup.target
* | |-local-fs.target
* | | |--.mount
* | | |--boot.mount
* | | |--rhel-import-state.service
* | | |--rhel-readonly.service
* | | `--systemd-remount-fs.service
* | `--swap.target
* | `--dev-mapper-centos_centosswap.swap
* `--timers.target
* `--systemd-tmpfiles-clean.timer
```

Listing 2.23 Auszug der Abhängigkeiten von »sshd«

Folgender Befehl wurde in diesem Abschnitt behandelt:

```
▶ systemctl list-dependencies <UNIT>
```

2.4.8 Logs mit journald

Wie bereits zu Beginn von Abschnitt 2.4 beschrieben wurde, bringt `systemd` sein eigenes Logging-Framework namens *journald* mit. Dass die Log-Dateien binär gespeichert werden, um sie länger und fälschungssicher – so zumindest der Anspruch der `systemd`-Entwickler – speichern zu können, ist jedoch ein großer Kritikpunkt der Linux-Community. Allerdings hat *journald* Charme und bringt außer der Umgewöhnung auch einige Vorteile mit, wie beispielsweise dass Fehler in den Log-Dateien in Rot markiert werden und so eher auffallen.

Rufen Sie beispielsweise `journalctl` ohne weitere Parameter auf, bekommen Sie einen interaktiven Auszug aller Log-Dateien, so wie sie früher in `/var/log/syslog` oder `/var/log/messages` landeten. Hier können Sie auch durch Eingabe eines großen »F« in den Follow-Modus wechseln. Mit dem Parameter `-f` oder `--follow` wird Ihnen das Log analog zu einem `tail -f` angezeigt. Wenn Sie die letzten 20 Log-Einträge anschauen wollen, benutzen Sie `-n 20` oder `--lines=20`. Der Parameter `--reverse` zeigt die Einträge in umgekehrter Reihenfolge an.

Einträge eines bestimmten Zeitraums grenzen Sie durch `--since` und `--until` ein. Dabei wird ein Datum in der Form "2018-07-30 18:17:16" ausgewertet. Ohne Datum wird der heutige Tag angenommen, ohne Sekunden wird 0 (null) angenommen, Sonderausdrücke wie `yesterday`, `today`, `tomorrow` oder `now` sind möglich.

Einer der wichtigsten Parameter ist `-u` oder `--unit=`, womit nur die Log-Dateien einer einzelnen Unit oder eines Satzes an Units ausgegeben werden. Wollen Sie beispielsweise die Log-Einträge des SSH-Daemons vom 5. Juni 2023 zwischen 13:00 Uhr und 14:00 Uhr haben, geben Sie den Befehl aus Listing 2.24 ein:

```
# journalctl --since="2023-06-05 13:00" --until="2023-06-05 14:00" \
--unit=sshd.service

-- Logs begin at Mo 2023-06-05 07:19:24 CEST, end at Mo 2023-06-05 15:56:51 CEST. --
Jun 05 13:07:24 centos sshd[13128]: reverse mapping checking getaddrinfo for \
                                1-2-3-4.a.b [1.2.3.4] failed - POSSIBLE BREAK-IN\
                                ATTEMPT!
Jun 05 13:07:24 centos sshd[13130]: reverse mapping checking getaddrinfo for \
                                1-2-3-4.a.b [1.2.3.4] failed - POSSIBLE BREAK-IN\
                                ATTEMPT!
Jun 05 13:07:24 centos sshd[13128]: Connection closed by 1.2.3.4 [preauth]
Jun 05 13:07:24 centos sshd[13130]: Connection closed by 1.2.3.4 [preauth]
```

Listing 2.24 Log-Auszug des SSH-Daemons

Die Logs von *journald* werden nach einem Neustart gelöscht. Wenn Sie das nicht wollen, sollten Sie das Verzeichnis `/var/log/journal` anlegen und das Signal `SIGUSR1` an den *journald*-Prozess senden. Damit werden die Logs in dem angegebenen Verzeichnis persistiert, sodass sie maximal zehn Prozent der Größe des Dateisystems belegen. Weitere Konfigurationen nehmen Sie in der Datei `/etc/systemd/journal.conf` vor.



Folgender Befehl wurde in diesem Abschnitt behandelt:

▶ `journalctl`

2.4.9 Abschlussbemerkung

Die hier vorgestellten Befehle und Direktiven bilden nur einen Ausschnitt der Möglichkeiten von `systemd` ab. Dieser Ausschnitt ist jedoch eine gute Basis für weitere Schritte. Eigene Skripte und Job-Definitionen können Sie damit bereits jetzt erstellen. Über die Manpages können Sie noch einige andere Kommandos und Subkommandos finden.

Kapitel 3

Festplatten und andere Devices

In diesem Kapitel geht es um Devices im weitesten Sinne. Dabei wird RAID genauso erklärt wie LVM. Außerdem wird ein zusätzliches Augenmerk auf »vdev« (das /proc-Dateisystem) sowie »udev« und die Erzeugung eigener udev-Regeln gelegt.

Für dieses Kapitel haben wir einige typische Anwendungsbeispiele von fortgeschrittenen Techniken rund um Festplatten und andere Devices herausgesucht. Wir erklären Ihnen, wie Sie Software-RAID verwenden können, und zeigen Ihnen, wie Sie den Logical Volume Manager (LVM) nutzen und wie Sie eigene *udev*-Regeln anlegen können. Abgeschlossen wird dieses Kapitel durch Erläuterungen zum /proc-Dateisystem.

3.1 RAID

Wenn im Folgenden von Geräten (englisch »Devices«) gesprochen wird, dann sind *Block-Devices* gemeint – also Geräte, auf die blockweise zugegriffen werden kann. Zur Gattung der Block-Devices gehören insbesondere Festplatten, USB-Sticks, aber auch *LUNs*, die von einem SAN (*Storage Area Network*) kommen.

Mit dem Begriff RAID (*Redundant Array of Independent Disks*) beschreibt man das Zusammenfassen von mehreren Geräten zu einem übergeordneten logischen Gerät. In der Regel fasst man Geräte zusammen, um größeren Speicherplatz zu bekommen, höhere Zugriffsgeschwindigkeiten zu erreichen oder um Ausfallsicherheit herzustellen.

RAIDs können sowohl durch die Hardware (spezielle RAID-Controller, die mehrere Festplatten zusammenfassen und dem Betriebssystem nur eine einzige logische Einheit melden) als auch durch Software erzeugt werden (aus mehreren Devices wird ein Special-Device erstellt). Verschiedene RAID-Level, die wir im Folgenden erklären werden, bieten spezifische Vor- und Nachteile.

Unerheblich ist, ob ein RAID-Controller eingesetzt oder das RAID durch Software verwaltet wird: Das Betriebssystem sieht in jedem Fall ein logisches Block-Device, mit dem es wie mit jeder anderen Festplatte verfahren kann. Die physischen Gerätschaften hinter dem logischen Laufwerk sind für das Betriebssystem nicht wichtig. Hardware-RAIDs sind schneller und robuster als Software-RAIDs.

3.1.1 RAID-0

RAID-0 wird auch *Striping* genannt. Bei diesem Verfahren werden zwei (oder mehr) Geräte zu einem einzigen zusammengefasst. Dadurch bekommt man den doppelten (oder mehrfachen) Speicherplatz. Das ist aber noch nicht alles: Die Soft- oder Hardware sorgt dafür, dass abwechselnd auf das eine und dann auf das andere Gerät geschrieben wird. So finden sich beispielsweise alle Blöcke mit ungerader Nummer auf Gerät eins und alle Blöcke mit gerader Nummer auf Gerät zwei. Bei drei oder mehr Geräten funktioniert das analog.

Mit der Anzahl der beteiligten Geräte steigt die Geschwindigkeit sowohl im Schreiben wie auch im Lesen. Nehmen wir an, Sie wollen 1 TB an Daten schreiben: Bei zwei Geräten würden 500 GB auf dem ersten Gerät landen und 500 GB auf dem zweiten. Wenn beide Geräte gleich schnell sind, würde damit die Transfergeschwindigkeit verdoppelt oder vermehrfacht werden. Allerdings führt der Ausfall eines Geräts dazu, dass die Daten nicht mehr lesbar sind. Der Einsatz von RAID-0 ist somit auf Bereiche beschränkt, bei denen die Performance besonders wichtig ist und die Daten im Fehlerfall schnell wiederherstellbar sind.

3.1.2 RAID-1

RAID-1 oder *Mirroring* (»Spiegelung«) wird auf fast allen Serversystemen zur Absicherung der Systemdaten benutzt. Die Spiegelung wird mit zwei oder mehr Geräten durchgeführt, die zu jeder Zeit alle einen identischen Inhalt haben. Alle Daten werden mehrfach geschrieben und sind somit auch beim Ausfall eines Geräts weiter verfügbar. Sollte ein Gerät ausfallen, so kann es im laufenden Betrieb ersetzt werden. Nachdem sich die Spiegelung synchronisiert hat, kann so weitergearbeitet werden, als wäre nichts ausgefallen. Beim Schreiben gibt es somit keine Performance-Steigerung. Ganz im Gegenteil: Die Geschwindigkeit des Schreibens ist abhängig vom langsamsten Gerät im RAID-1-Verbund. Einzig beim Lesen kann es mit guten Controllern oder guter Software zu einer Geschwindigkeitssteigerung kommen, wenn wie bei RAID-0 wechselseitig gelesen wird. Man spricht in dem Fall auch von *RAID 0+1*.

Der große Nachteil dieses Verfahrens ist, dass der Speicherplatz der Geräte nicht komplett genutzt wird. Zwei Geräte mit 1 TB erzeugen ein logisches Laufwerk mit 1 TB nutzbarem Platz, nicht mehr. Ein *RAID-1* schützt nicht vor logischen Fehlern und ist daher nicht mit einem Backup zu verwechseln. Löschen Sie eine Datei, ist sie unwiederbringlich verschwunden.

3.1.3 RAID-5

Für ein *RAID-5* werden wenigstens drei Geräte benötigt. Ähnlich wie bei RAID-0 werden die Daten auf zwei Geräte verteilt, das dritte Gerät enthält eine Prüfsumme (auch *Parität* genannt) – in der Regel werden die Daten des ersten und zweiten Geräts mit *XOR* verknüpft – und kann so beim Ausfall eines Geräts den Inhalt wieder zurückrechnen. Tabelle 3.1 gibt Ihnen einen Eindruck, wie das funktioniert:

Gerät 1	Gerät 2	Gerät 3
Datenblock 1	Datenblock 2	Prüfsumme 1/2
Prüfsumme 3/4	Datenblock 3	Datenblock 4
Datenblock 5	Prüfsumme 5/6	Datenblock 6
Datenblock 1	Datenblock 2	Prüfsumme 1/2
...

Tabelle 3.1 RAID-5

RAID-5 kombiniert zum Teil die Vorteile eines RAID-0 mit einer Ausfallsicherheit. Wenn ein einzelnes Gerät ausfällt, sind alle Daten noch vorhanden, und durch Austausch des defekten Geräts kann die Redundanz wiederhergestellt werden. Da immer eine Festplatte mit Prüfsummen belegt ist, ist die Kapazität eines RAID-5-Verbunds bestimmt durch die Anzahl der Geräte minus eins.

Üblich sind RAID-5-Konfigurationen mit vier Geräten. Auf drei Geräten finden sich Daten und auf dem vierten Gerät die Prüfsummen analog zu Tabelle 3.1. Der Geschwindigkeitsvorteil beim Schreiben der Daten ist leider gering, da die Paritätsinformationen bei jedem Schreibzugriff neu berechnet werden müssen. Dafür steigt die Lesegeschwindigkeit mit der Anzahl der verwendeten Geräte.

3.1.4 RAID-6

Bei RAID-6-Systemen werden zwei Geräte für Prüfsummen verwendet und nicht nur ein Gerät wie bei RAID-5. Hier dürfen zwei Geräte ausfallen, bevor Daten verloren gehen. Da die Schreibgeschwindigkeit gegenüber RAID-5 noch schlechter ist, wird dieses Verfahren nur sehr selten angewendet. Üblicher ist es, ein nicht benutztes Gerät zusätzlich zu verwenden (*Hot Spare* genannt), um im Fehlerfall das defekte Gerät zu ersetzen.

3.1.5 RAID-10

RAID-10 – gesprochen »eins-null«, nicht »zehn« – ist eine Kombination aus RAID-0 und RAID-1. Zwei Geräte werden mittels RAID-1 gespiegelt und eine oder mehrere dieser Spiegelungen wird bzw. werden mit einem RAID-0 zusammengefasst (»gestripet«). Damit bietet RAID-10 eine sehr gute Performance, aber leider auch den Nachteil, dass nur die Hälfte der Gesamtkapazität verwendet werden kann. Dafür darf in jedem Fall ein Gerät ausfallen, und im gleichen *Stripeset* darf sogar ein zweites Gerät ausfallen.

3.1.6 Zusammenfassung

In Tabelle 3.2 sehen Sie eine Übersicht über die RAID-Systeme und deren Performance.

RAID-Level	Benötigte Geräte (n)	Nettokapazität	Lesepformance	Schreibperformance
0	mindestens 2	$n \times \text{Größe}$	n	n
1	mindestens 2	$(n/2) \times \text{Größe}$	2/n oder n	n/2
5	mindestens 3	$(n-1) \times \text{Größe}$	n	n/4
6	mindestens 4	$(n-2) \times \text{Größe}$	n	n/6
10	mindestens 4	$(n/2) \times \text{Größe}$	n/2 oder n	n/2

Tabelle 3.2 Übersicht der RAID-Level

Im Regelfall werden bei produktiven Servern die Geräte, auf denen das System läuft, mit RAID-1 gespiegelt. So führt der Ausfall eines Geräts nicht zum Ausfall des Servers, und es gehen keine Daten verloren. Bei großen Datenmengen – wie beispielsweise bei Backupservern oder Bilddatenbanken – wird ein RAID-5 verwendet, weil der »Verschnitt« nicht so groß und eine Ausfallsicherung gegeben ist. Das wird aber mit einer Reduktion der Schreibgeschwindigkeit erkaufte. Hardware-RAID-Controller, die die Paritätsberechnung per Hardware implementiert haben, können diesen Nachteil aufwiegen.

Wenn Sie hingegen mit einem Software-RAID arbeiten, kann die verminderte Schreibgeschwindigkeit zu Problemen führen. So können beispielsweise Backups abbrechen, weil zu viele Server gleichzeitig gesichert werden. Einen Spezialfall bilden RAID-5-Systeme mit sehr vielen Geräten: Dann nimmt die Performance wieder zu.

Sollten Sie viel Speicherplatz und hohe Performance sowie Ausfallsicherheit benötigen, führt kaum etwas an einem RAID-10 oder an einem *Storage Area Network* (SAN) vorbei.




RAID-Systeme je nach Zweck auswählen!

Wie Sie gesehen haben, haben sowohl die Anzahl der Festplatten wie auch der gewählte RAID-Level einen Einfluss auf die Lese- und Schreibperformance. So kann es sinnvoller sein, ein RAID-System mit vielen kleineren Festplatten zu betreiben, als den Speicherplatz durch wenige große Festplatten zur Verfügung stellen zu lassen (wenn Sie sich für ein RAID-5 entscheiden). RAID-5 mit drei identischen Festplatten erreicht gemäß Tabelle 3.2 nur drei Viertel der Geschwindigkeit einer einzelnen Festplatte. Mit sechs identischen Festplatten sind Sie bereits bei der 1,5-fachen Geschwindigkeit, und selbst das könnte – je nach Anwendungszweck – zu langsam sein.

Sollten Sie beispielsweise die vierfache Schreibgeschwindigkeit einer einfachen Festplatte benötigen, so können Sie diese mit 16 Festplatten im RAID-5-Verbund oder mit acht Festplatten in einem RAID-10 oder mit vier Festplatten (ohne Ausfallsicherung) in einem RAID-0 aufbauen.

Nicht nur der RAID-Level, auch und vor allem die Anzahl der beteiligten Festplatten haben einen Einfluss auf die Lese- und Schreibperformance. Es gibt leider kein System, das allen Anforderungen gleichermaßen gerecht wird.

3.1.7 Weich, aber gut: Software-RAID

Eine Anmerkung vorab: Wenn sich Ihnen die Möglichkeit der Entscheidung bietet, sollten Sie einen Hardware-RAID-Controller immer einem Software-RAID vorziehen. Die Hardware bietet spezialisierte Prozessoren und Chips, die die RAID-Operationen – Paritätsberechnung und Verteilung der Lese- und Schreibzugriffe – schneller durchführen als ein Allzweckprozessor in Ihrem Server. 

In einem Software-RAID wird diese Aufgabe vom Hauptprozessor übernommen. Da heutige CPUs leistungsfähig genug sind, ist das im normalen Betrieb kein großes Problem. Wenn allerdings ein Gerät ausfällt, kommt es zu einer erheblichen Mehrlast, insbesondere dann, wenn die Geräte mit *RAID-5* verbunden sind. Schließlich muss für jeden Block auf den beteiligten Geräten eine neue Prüfsumme berechnet und die Verteilung der Daten neu angestoßen werden. Die CPU-Last kommt zusätzlich zur I/O-Last auf die Geräte, was dazu führen kann, dass der Computer nicht mehr benutzbar ist und enorm träge reagiert. Wenn es also außer auf die Verfügbarkeit auch auf die Möglichkeit einer ressourcenschonenderen Wiederherstellung nach einem Fehlerfall ankommt, dann sind Sie mit einem Software-RAID nicht so gut bedient wie mit einem Hardware-RAID. Klar ist leider, dass auch mit einem Hardware-RAID die Performance sinkt, wenn ein Neuaufbau des RAIDs läuft, aber der Einbruch ist nicht ganz so drastisch wie bei der Software.

Ein weiterer Punkt, der für ein Hardware-RAID spricht, ist die eingebaute Batterie. Durch sie kann bei einem Stromausfall der komplette Inhalt des Caches noch auf die Festplatten geschrieben werden, und die ausstehenden Transaktionen können abgeschlossen werden. Aus Performance-Sicht bringt das auch einen Vorteil: Der Controller ist in der Lage, eine I/O-Operation als abgeschlossen zu melden, wenn sie sich im Cache des Controllers befindet, und nicht erst dann, wenn sie bereits auf der Festplatte ist. Software-RAIDs sind enorm anfällig für Stromausfälle und reagieren bestenfalls mit einem Neuaufbau des *RAID-Arrays* nach dem Ausfall. Im schlimmsten Fall sind die Daten zwischen den einzelnen Geräten eines RAIDs nicht mehr konsistent und es droht Datenverlust.

Aus den genannten Gründen sind RAID-Controller und damit Hardware-RAIDs bei Festplatten vorzuziehen, wenn es das Budget erlaubt. Für kleinere Systeme und auch für Geräte, die

aus dem SAN angebunden sind, sind Software-RAIDs eine gute Option – im Fall von SAN sogar der beste Weg, wenn nicht das SAN die Spiegelung übernimmt.



Nicht alles, was sich RAID-Controller nennt, ist auch einer. Als Faustregel gilt: Wenn Sie unter Linux noch einzelne Festplatten und keine logischen Laufwerke sehen, stehen die Chancen gut, dass der RAID-Controller keiner ist. Diese falschen Controller werden auch *FakeRAID*-Controller genannt. In diesem Fall greift keiner der oben genannten Vorteile eines Hardware-RAID-Controllers. Aber auch, wenn Sie logische Laufwerke sehen, kann es sein, dass der Hauptprozessor Ihres Systems die Hauptarbeit leistet, da Linux eventuell einen *Fake-RAID*-Treiber einsetzt, der die Arbeit übernimmt. Bitte informieren Sie sich gut vor dem Kauf, um eventuelle Enttäuschungen zu vermeiden.

3.1.8 Software-RAID unter Linux

Nach den Vorbetrachtungen sind wir nun endlich so weit, um Software-RAIDs unter Linux aufzubauen und zu verwalten. In diesem Abschnitt geht es um das Tool *mdadm*; beachten Sie bitte, dass Sie auch mit dem *Logical Volume Manager (LVM)* (siehe Abschnitt 3.2.9, »Mirroring ausführlich«) Software-RAIDs verwalten können.

Bitte nutzen Sie die Möglichkeiten der Installationsroutinen Ihrer Distribution, wenn Sie Ihr System auf gespiegelten Systemplatten betreiben wollen. Auf den folgenden Seiten geht es um den Aufbau von RAIDs in einem laufenden System, um ihre Verwaltung und um die Wiederherstellung nach einem Fehlerfall. Bitte installieren Sie das Paket *mdadm* mit den Mitteln Ihrer Distribution.

Im virtuellen */proc*-Dateisystem (siehe Abschnitt 3.4, »Alles virtuell? */proc*«) sehen Sie den Status aller Gerätschaften unter der Kontrolle von *mdadm* unter */proc/mdstat*. Wenn Sie noch kein RAID definiert haben, können Sie aber zumindest die unterstützten Features (»Personalities«) Ihrer *mdadm*-Version sehen. Listing 3.1 zeigt Ihnen die Ausgabe eines minimalen openSUSE-Systems:

```
opensuse:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [raid10]
unused devices: <none>
```

Listing 3.1 »mdstat« ohne konfigurierte RAIDs

Für den Fall, dass das Kommando aus Listing 3.1 keine sinnvolle Ausgabe bietet, müssen Sie via *modprobe* eines oder mehrere der Kernelmodule *raid0*, *raid1*, *raid456*, *raid6* oder *raid10* quasi »von Hand« laden. Mit dem folgenden Kommando legen wir ein RAID-5-System mit drei Geräten an:

```
opensuse:~ # mdadm --create --verbose /dev/md0 --level=raid5 \
--raid-devices=3 /dev/sdb /dev/sdc /dev/sdd
```

```
mdadm: layout defaults to left-symmetric
mdadm: layout defaults to left-symmetric
mdadm: chunk size defaults to 512K
mdadm: size set to 1047552K
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
```

Listing 3.2 RAID-5 mit drei Festplatten

Glücklicherweise sind die Optionen sprechend. Ein neues RAID wird mit `--create` erstellt, und die Art des neuen RAID wird mit `--level` angegeben. Da sich die RAIDs je nach Anzahl der aktiv verwendeten Geräte unterscheiden und unterschiedlich erzeugt werden, muss diese Anzahl auch noch dem Programm mit `--raid-devices` mitgeteilt werden. Sie werden in Listing 3.15 noch sehen, dass dieser Parameter wichtig ist. Für eine ausführliche Ausgabe sorgt der Parameter `--verbose`. Der Name des neuen Geräts (oder »logischen Laufwerks«) ist `/dev/md0`, wobei sich die Abkürzung `md` auf »Multiple Devices« bezieht. Verschiedene Faktoren beeinflussen die Dauer des Aufbaus; mit `cat /proc/mdstat` können Sie jederzeit – auch während des Aufbaus – den aktuellen Status überprüfen.

```
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sdd[3] sdc[1] sdb[0]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [UU_]
      [>.....] recovery = 3.8% (40320/1047552) \
      finish=2.9min speed=5760K/sec
[...]
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sdd[3] sdc[1] sdb[0]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU]
[...]

```

Listing 3.3 Der RAID-Status aus »/proc/mdstat«

Wie Sie an der Ausgabe in Listing 3.3 feststellen können, ist der Aufbau von `md0` erfolgreich gewesen. `[UUU]` zeigt den Status der beteiligten Geräte an. `U` steht dabei für »Up«, im Fehlerfall oder beim Neuaufbau des RAID würden Sie einen Unterstrich `_` für »Down« sehen. Die Ziffern `[3/3]` bedeuten, dass das Gerät im Idealfall aus 3 Einzelteilen besteht (die Ziffer vor dem Schrägstrich) und dass alle drei Geräte gerade aktiv sind (die Ziffer nach dem Schrägstrich). Kein Grund zur Sorge besteht, wenn die zweite Ziffer größer oder gleich der ersten Ziffer ist. Nun können wir das Gerät formatieren und einbinden:

```
opensuse:~ # mkfs -t ext4 /dev/md0
mke2fs 1.42.8 (20-Jun-2013)
```

```
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=128 blocks, Stripe width=256 blocks
131072 inodes, 523776 blocks
26188 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=536870912
16 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

```
opensuse:~ # mount /dev/md0 /mnt
opensuse:~ # df -h /mnt
Filesystem      Size  Used Avail Use% Mounted on
/dev/md0        2.0G  3.0M  1.9G   1% /mnt
```

Listing 3.4 Formatieren und Mounten des neuen RAID-Systems

Die bisher getroffenen Einstellungen überstehen den nächsten Neustart des Systems nicht. Um die Konfiguration zu sichern und das RAID nach einem Reboot zur Verfügung zu haben, muss sie noch mit `mdadm` in die Datei `/etc/mdadm.conf` geschrieben werden (siehe Listing 3.5).




Bitte beachten Sie, dass sich der Speicherort je nach Distribution unterscheidet: Bei CentOS und openSUSE befindet er sich in der Datei `/etc/mdadm.conf` und bei Ubuntu/Debian in `/etc/mdadm/mdadm.conf`.

```
opensuse:~# mdadm --examine --scan >> /etc/mdadm.conf
opensuse:~# cat /etc/mdadm.conf
ARRAY /dev/md/0 metadata=1.2 \
UUID=c23aefab:abd8738f:b031f231:60131eba name=opensuse:0
```

Listing 3.5 Konfiguration des neuen RAID-Systems

Bei jedem Neustart überprüft `mdadm` alle Partitionen und versucht Konfigurationsdaten zu finden. Wenn es die Daten findet, ist das Tool in der Lage, die RAIDs auch eigenständig wieder aufzubauen.

Sollten die Konfigurationen allerdings korrupt sein, kommen Sie ohne `/etc/mdadm.conf` nicht weiter. Der schlimmstmögliche Fall wäre, dass die Konfigurationsdateien nicht wiederherstellbar sind. Um dies auszuschließen, sollten Sie die Konfiguration auf jeden Fall schreiben.

Bei Debian und Ubuntu ändert sich der Device-Name nach einem Neustart; in unserem Test wurde aus `/dev/md0` ein `/dev/md127`. Sie können jederzeit den Namen mit einem Blick auf `/proc/mdstat` überprüfen. 

Mit einer zusätzlichen nicht benutzten Festplatte können Sie die Datensicherheit erhöhen, wenn Sie diese zum RAID hinzufügen. Nicht benutzte Festplatten, die nicht aktiv verwendet werden, werden *Spare Disks* oder *Hot Spares* genannt. Falls in einem RAID eine Festplatte ausfällt, übernimmt die *Hot-Spare-Disk*, und das RAID baut sich mit dieser neu auf. In Listing 3.6 sehen Sie, wie Sie vorgehen müssen:

```
opensuse:~ # mdadm --add /dev/md0 /dev/sde
mdadm: added /dev/sde

opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sde[4](S) sdb[0] sdc[1] sdd[3]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU] [...]
```

Listing 3.6 »Spare Disk« hinzufügen

Das »(s)« hinter dem Festplattennamen – hier `/dev/sde` – steht für »Spare Disk.« In Listing 3.7 sehen Sie, dass die Spare Disk übernimmt, wenn eine Festplatte ausfällt:

```
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sdd[3](F) sdc[1] sde[4] sdb[0]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [UU_]
      [==>.....] recovery = 17.3% (182280/1047552) finish=2.7min \
      speed=5312K/sec [...]
```

Listing 3.7 Ausfall von »/dev/sdd«

Die defekte Festplatte muss ausgetauscht werden. Wenn Ihr System das hardwaretechnisch unterstützt, funktioniert das auch online. In Listing 3.8 sehen Sie, wie die defekte Festplatte entfernt wird. Um das tun zu können, muss der Controller der Festplatte noch reagieren:

```
opensuse:~ # mdadm --manage --replace /dev/md0 /dev/sdd
mdadm: Marked /dev/sdd (device 2 in /dev/md0) for replacement

opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
```

```
md0 : active raid5 sdb[0] sde[4] sdd[3](F) sdc[1]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU] [...]
```

Listing 3.8 Entfernen der defekten Festplatte



Bitte beachten Sie, dass neuere Versionen des Software-RAIDs die Disk automatisch aus dem RAID-Verbund entfernen, wenn sie nicht mehr ansprechbar ist. Der Zwischenschritt aus Listing 3.8 ist dann nicht nötig.

Wenn Sie nun so, wie in Listing 3.6 beschrieben, die Festplatte neu hinzufügen, wird sie als neue Spare Disk verwendet. Falls das RAID einen größeren Fehler hat, schafft es die RAID-Software `mdadm` nicht mehr, selbstständig das RAID wieder aufzubauen (siehe Listing 3.9). In solchen Fällen müssen Sie selbst Hand anlegen, wenn Sie das RAID wieder benutzen wollen.

```
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active (auto-read-only) raid5 sdd[3](S) sdc[1] sde[4]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [_UU] [...]
```

Listing 3.9 Defektes RAID



Bitte bewahren Sie Ruhe, und überlegen Sie genau Ihre nächsten Schritte. Es besteht die Möglichkeit, dass Sie Fehler machen, die Ihre Daten unwiederbringlich löschen. Wenn Sie kein Backup Ihrer Daten haben, wäre jetzt eine gute Gelegenheit, mittels *Disk Images* die Rohdaten Ihrer Festplatten zu sichern.

Wie das genau funktioniert, können Sie in Abschnitt 7.5.3, »Erstellen eines Images mit `dd`«, nachlesen. Für die nächsten Schritte ist es wichtig, zuerst das RAID-System zu deaktivieren:

```
opensuse:~ # mdadm --manage --stop /dev/md0
mdadm: stopped /dev/md0
```

```
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
unused devices: <none>
```

Listing 3.10 Deaktivierung des RAIDs

Jetzt können Sie mit der Analyse beginnen. Wie Sie in Listing 3.11 sehen können, scheint die erste Festplatte des RAIDs keinen Block mit Konfigurationsdaten mehr zu haben:

```
opensuse:~ # mdadm --misc --examine /dev/sdb
mdadm: No md superblock detected on /dev/sdb.
```

```
opensuse:~ # mdadm --misc --examine /dev/sdc
/dev/sdc:
```



```

    Magic : a92b4efc
    Version : 1.2
    Feature Map : 0x0
    Array UUID : c23aefab:abd8738f:b031f231:60131eba
        Name : opensuse:0 (local to host opensuse)
    Creation Time : Sat Aug 16 15:00:56 2014
    Raid Level : raid5
    Raid Devices : 3

    Avail Dev Size : 2095104 (1023.17 MiB 1072.69 MB)
        Array Size : 2095104 (2046.34 MiB 2145.39 MB)
        Data Offset : 2048 sectors
        Super Offset : 8 sectors
        Unused Space : before=1960 sectors, after=0 sectors
            State : clean
        Device UUID : ddb975d5:3fe1883e:bd214593:623a5fe5

    Update Time : Sat Aug 16 16:05:19 2014
    Bad Block Log : 512 entries available at offset 72 sectors
        Checksum : d9036ccf - correct
        Events : 49

    Layout : left-symmetric
    Chunk Size : 512K

    Device Role : Active device 1
    Array State : AAA ('A' == active, '.' == missing, 'R' == replacing)

```

Listing 3.11 Detailuntersuchung von »/dev/sdb«

Auf der zweiten Festplatte scheinen aber noch alle Daten vorhanden zu sein. Der Block der dritten Festplatte ähnelt dem der zweiten und ist aus diesem Grund hier nicht aufgeführt.

In Listing 3.11 finden Sie alle Daten, die das RAID betreffen, und im unteren Teil bei Array State sehen Sie den letzten bekannten Zustand des RAIDs. In unserem Fall – das RAID war read-only vorhanden – konnte der Zustand nicht mehr gespeichert werden. Wichtig ist, dass die Array UUID bei allen am RAID beteiligten Festplatten identisch ist.

Wir hatten ein RAID-5 gebaut und dieses mit drei Festplatten versehen. Zwei von den beteiligten Festplatten scheinen noch intakt zu sein, und wir haben in diesem Fall sogar eine Spare Disk. Das bedeutet in Summe, dass wir in der Lage sind, das RAID so wie in Listing 3.12 wieder aufzubauen, ohne dass wir Datenverlust befürchten müssen:

```

opensuse:~ # mdadm --assemble --run /dev/md0 /dev/sdc /dev/sdd /dev/sde
mdadm: /dev/md0 has been started with 2 drives (out of 3) and 1 spare.

```

```
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sdc[1] sdd[3] sde[4]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [_UU]
      [==>.....] recovery = 15.4% (162664/1047552) [...]
```

Listing 3.12 Zusammenbauen des RAID-Verbunds

Zur Sicherheit sollten Sie noch das Dateisystem prüfen. Prinzipiell sollte es keine Probleme gegeben haben. Allerdings wurde das Filesystem in unserem Fall unsauber geschlossen:

```
opensuse:~ # fsck /dev/md0
fsck from util-linux 2.23.2
e2fsck 1.42.8 (20-Jun-2013)
/dev/md0 contains a file system with errors, check forced.
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/md0: 11/131072 files (0.0% non-contiguous), 17196/523776 blocks
```

Listing 3.13 Dateisystemprüfung

Wenn der Superblock mit den RAID-Konfigurationsdaten noch in Ordnung gewesen wäre, hätten Sie die defekte Festplatte wieder dem RAID hinzufügen können. So bleibt Ihnen nur das Hinzufügen einer neuen Festplatte als Spare Disk:

```
opensuse:~ # mdadm --manage /dev/md0 --re-add /dev/sdb
mdadm: --re-add for /dev/sdb to /dev/md0 is not possible
opensuse:~ # mdadm --manage /dev/md0 --add /dev/sdb
mdadm: added /dev/sdb
opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sdb[5](S) sdc[1] sdd[3] sde[4]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU] [...]
```

Listing 3.14 Hinzufügen einer neuen Festplatte

RAIDs können auch vergrößert werden. Sie müssen nur dafür sorgen, dass eine als Spare Disk hinzugefügte Festplatte vom RAID aktiv benutzt wird. Dazu sind zwei Optionen notwendig: `--grow` sorgt für eine Vergrößerung des RAIDs, und zusammen mit `--raid-devices` bestimmen Sie, aus wie vielen Geräten das RAID zusammengesetzt werden soll:

```
opensuse:~ # mdadm --grow /dev/md0 --raid-devices=4
mdadm: Need to backup 3072K of critical section..
```

```

opensuse:~ # cat /proc/mdstat
Personalities : [linear] [raid6] [raid5] [raid4]
md0 : active raid5 sdb[5] sdc[1] sdd[3] sde[4]
      2095104 blocks super 1.2 level 5, 512k chunk, algorithm 2 [4/4] [UUUU]
      [>.....] reshape = 0.8% (9220/1047552) \
      finish=4.4min speed=3842K/sec [...]

```

Listing 3.15 Vergrößern eines RAID-Verbunds

Sobald das Umarrangieren der Daten – in `/proc/mdstat` *reshape* genannt – abgeschlossen ist, sollten Sie noch das Dateisystem vergrößern:

```

opensuse:~ # df -h /mnt
Filesystem      Size  Used Avail Use% Mounted on
/dev/md0        2.0G  3.0M  1.9G   1% /mnt

opensuse:~ # resize2fs /dev/md0
resize2fs 1.42.8 (20-Jun-2013)
Filesystem at /dev/md0 is mounted on /mnt; on-line resizing required
old_desc_blocks = 1, new_desc_blocks = 1
The filesystem on /dev/md0 is now 785664 blocks long.


```

```

opensuse:~ # df -h /mnt
Filesystem      Size  Used Avail Use% Mounted on
/dev/md0        3.0G  3.0M  2.8G   1% /mnt

```

Listing 3.16 Anpassen des Dateisystems

Das Umarrangieren oder der *Rebuild* der Daten ist sehr zeitintensiv und darf nicht unterbrochen werden. In dieser Zeit ist die Performance sehr eingeschränkt. 

Die in diesem Abschnitt gezeigten Beispiele wurden mit einer virtuellen Maschine unter KVM erstellt. Die RAID-Devices waren nur jeweils 1 GB groß und lagen auf einer SSD. Das Filesystem war bis auf einige Bytes komplett leer. Die Erweiterung des RAID von drei auf vier aktive Festplatten hat dennoch etwa zehn Minuten gedauert. Dass die Laufzeiten bei »echten« und großen Festplatten im Bereich mehrerer Tage liegen, ist keine Seltenheit.

3.1.9 Abschlussbemerkung zu RAIDs

Die Verwendung von RAID-Systemen gehört zu einem verlässlichen Serverbetrieb dazu. Allerdings müssen diese RAIDs auch überwacht werden, um nicht eines Tages vor einem Datenverlust zu stehen. Idealerweise nehmen Sie den Status des RAID in Ihre Monitoring-Lösung mit auf. Wie Sie Ihr eigenes Monitoring einrichten können, haben wir für Sie in Kapitel 29, »Monitoring – wissen, was läuft«, ausführlich beschrieben.

3.2 Rein logisch: Logical Volume Manager (LVM)

Partitionierungen auf Linux-Systemen sind relativ starr. Es ist nicht möglich, nach einer begonnenen Installation die einzelnen beteiligten Geräte zu vergrößern oder zu verkleinern. (SAN-Speicher bildet hier eine Ausnahme.) Eine einmal getroffene Entscheidung für das Layout lässt sich im Nachhinein nur noch dadurch ändern, dass weitere Festplatten oder – allgemeiner gesagt – Geräte irgendwo in den Verzeichnisbaum eingehängt werden.



Nehmen wir an, wir hätten eine Festplatte von 50 GB, die so wie in Tabelle 3.3 aufgebaut ist (das Beispiel vereinfacht die Grundsituation, um das Prinzip zu erläutern).

Device	Größe	Beschreibung
/dev/sda	50 GB	die gesamte Festplatte
/dev/sda1	10 GB	Mountpunkt / (das System)
/dev/sda2	36 GB	Mountpunkt /home (die Homeverzeichnisse)
/dev/sda5	4 GB	Swap

Tabelle 3.3 Beispiel-Layout ohne LVM

Im Laufe des Lebens dieser Maschine stellt sich heraus, dass der Mountpunkt mit den Homeverzeichnissen vergrößert werden muss, weil es noch weitere Nutzer gibt, die das System benutzen sollen. Eine weitere Festplatte mit 50 GB wird angeschafft, und für die neuen User werden die neuen Homeverzeichnisse als separate Partitionen erstellt.

Device	Größe	Beschreibung
/dev/sda	50 GB	die gesamte Festplatte
/dev/sda1	10 GB	Mountpunkt / (das System)
/dev/sda2	36 GB	Mountpunkt /home (die Homeverzeichnisse)
/dev/sda5	4 GB	Swap
/dev/sdb	50 GB	die neue Festplatte
/dev/sdb5	5 GB	Mountpunkt /home/neuernutzer1
/dev/sdb6	5 GB	Mountpunkt /home/neuernutzer2

Tabelle 3.4 Beispiel-Layout ohne LVM mit neuer Festplatte

Wie Sie in Tabelle 3.4 feststellen können, ist klar abzusehen, dass nach spätestens zehn neuen Nutzern diese Vorgehensweise an ihre Grenzen stößt – und das, obwohl eventuell die Verzeichnisse der neuen Benutzer gar nicht oder nur sehr gering gefüllt sind. Alternativ können Sie natürlich ein Backup des kompletten Systems erstellen, eine größere primäre Festplatte anschaffen und das Backup auf die neue Festplatte übertragen, wobei Sie die Partitionsgrößen anpassen. Dieses Vorgehen erfordert allerdings ein längeres Wartungsfenster, in dem das System nicht verfügbar ist.

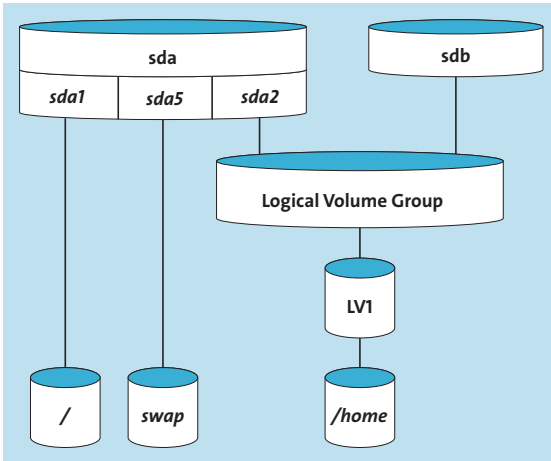


Abbildung 3.1 Das neue Layout, grafisch dargestellt

Wenn aber beim ursprünglichen Design des Layouts LVM einbezogen wird, bekommen Sie die Möglichkeit geschenkt, auch im Nachhinein noch Spiegelungen und Erweiterungen aufzubauen und verwalten zu können. Wenn jetzt der Platz zur Neige geht, können Sie, wie in Abbildung 3.1 gezeigt, auf die *Logical Volume Group* ausweichen. Mit der gleichen Vorbedingung wie in Tabelle 3.4 kann man den Platz, den *sda2* bietet, einer sogenannten *Logical Volume Group* zuweisen, sodass das Beispiel-Layout so wie in Abbildung 3.1 oder Tabelle 3.5 aussieht.

Device	Größe	Beschreibung
/dev/sda	50 GB	die gesamte Festplatte
/dev/sda1	10 GB	Mountpunkt / (das System)
/dev/vg1/lv1	36 GB	Mountpunkt /home (die Homeverzeichnisse)
/dev/sda5	4 GB	Swap

Tabelle 3.5 Beispiel-Layout mit LVM

Es ist an dieser Stelle wichtig, zu erwähnen, dass alle modernen Filesysteme (mehr zu Dateisystemen finden Sie in Kapitel 4, »Dateisysteme«) in der Größe veränderbar sind und dass nicht der gesamte Platz den Homeverzeichnissen zugewiesen werden muss. Das Beispiel vereinfacht die Situation stark, um das Prinzip zu verdeutlichen (siehe Tabelle 3.6).

Device	Größe	Beschreibung
/dev/sda	50 GB	die gesamte Festplatte
/dev/sda1	10 GB	Mountpunkt / (das System)
/dev/vg1/lv1	86 GB	Mountpunkt /home (die Homeverzeichnisse)
/dev/sda5	4 GB	Swap
/dev/sdb	50 GB	neue Festplatte; der Platz kommt der Logical Volume Group zugute.

Tabelle 3.6 Beispiel-Layout mit LVM und mit neuer Festplatte

3.2.1 Grundlagen und Begriffe

Um mit LVM sicher umgehen zu können, müssen Sie einige Begriffe beherrschen.

LVM-Begriffe

Das sind die wichtigsten Vokabeln und Abkürzungen, die Sie im Umgang mit LVM benötigen:

- ▶ **Volume Group (VG):** *Volumengruppe* oder *Diskgruppe*, die Speicherplatz für Devices verwaltet
- ▶ **Physical Volume (PV):** ein Festplattenbereich oder eine Partition oder eine LUN, die einer Volume Group zur Verfügung gestellt wird
- ▶ **Physical Extent (PE):** kleinste Verwaltungseinheit eines Physical Volumes, die global für eine Volume Group festgelegt wird
- ▶ **Logical Extent (LE):** die zum *Physical Extent* passende und gleich große logische Verwaltungseinheit; sie verweist auf einen *Physical Extent*.
- ▶ **Logical Volume (LV):** *logisches Volumen* oder *Disk*, auf die ein Dateisystem aufgebracht werden kann



Bitte beachten Sie, dass die Begriffe bei anderen Volume Managern anders verwendet werden können.

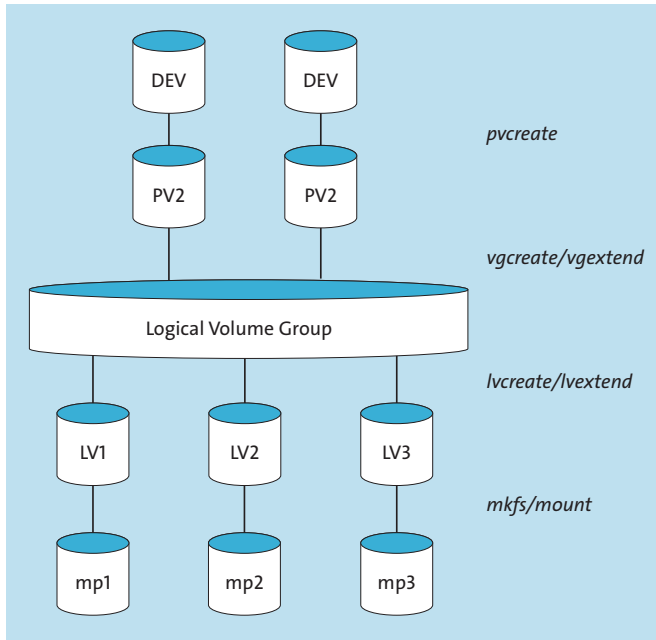


Abbildung 3.2 Übersicht über die Struktur von LVM

Wenn von *Disks* die Rede ist, kann es sich sowohl um Festplatten als auch um USB-Speicher oder LUNs von SAN-Systemen – wir sprechen hier auch von *Block-Devices* – handeln. LVM setzt auf die physische Schicht, also auf die Disks, eine logische Verwaltungsschicht. Das hat den Vorteil, dass man an den Disks Veränderungen vornehmen kann, ohne dass die logische Schicht davon beeinträchtigt wird. Abbildung 3.2 gibt Ihnen einen groben Überblick darüber, wie die Begriffe zusammenhängen (»mp« steht dabei für Mountpunkt).

3.2.2 Setup

In diesem Abschnitt befassen wir uns mit der Installation des Logical Volume Managers. In allen von uns beschriebenen Distributionen müssen Sie das Paket »lvm2« installieren, um den Logical Volume Manager nutzen zu können.

Nach der Installation gibt es ein großes Bündel von neuen Programmen:

vgcfbackup	vgconvert	vgextend	vgmknodes	vgs
vgcfrestore	vgcreate	vgimport	vgreduce	vgscan
vgchange	vgdisplay	vgimportclone	vgremove	vgsplit
vgck	vgexport	vgmerge	vgrename	

pvchange	pvcreate	pvmove	pvresize	pvscan
pvck	pvdiskdisplay	pvremove	pvs	

lvchange	lvdisplay	lvnconfig	lvmpolld	lvreduce	lvresize
lvconvert	lvextend	lvmdiskscan	lvmsadc	lvremove	lvs
lvcreate	lvm	lvmdump	lvmsar	lvrename	lvscan

Listing 3.17 Programme, die mit LVM installiert werden

Es sei an dieser Stelle darauf hingewiesen, dass bei LVM – anders als bei anderen Volume Managern – die Benennung der Befehle über die einzelnen Ebenen hinweg konsistent ist.

Das Erzeugen funktioniert mit `create` und der »Vorsilbe« `pv` für Physical Volumes, `vg` für Volume Groups, `lv` für Logical Volumes. Gleiches gilt für viele andere Befehle.

3.2.3 Aufbau einer Volume Group mit einem Volume

In diesem Beispiel nehmen wir einmal an, dass wir ein System mit einer Festplatte haben und diese so partitionieren wollen, wie es in Tabelle 3.5 beschrieben wurde.

Wenn das Partitionstool selbst keine Erzeugung von LVs zulässt, legen wir bei der Installation eine *root*-Partition an (beispielsweise `/dev/sda1`). Auf die Erstellung einer *swap*-Partition können wir für dieses Beispiel verzichten. Zum Schluss erzeugen wir eine Partition (beispielsweise `/dev/sda2`), der wir noch kein Dateisystem zuweisen. Sollte das Anlegen einer leeren Partition nicht möglich sein, lassen wir den Platz unkonfiguriert und legen nach erfolgter Installation mittels `fdisk /dev/sda` die leere Partition `/dev/sda2` von Hand an.

Um diese Partition – es könnte auch durchaus eine komplette Festplatte sein – zur Benutzung durch LVM vorzubereiten, führen wir den Befehl `pvcreate /dev/sda2` aus. `pvcreate` erzeugt einige Daten im Header der Partition, die LVM benötigt, um die Partition ansprechen und verwalten zu können. Die Anwendung des Befehls zeigt auch schon, dass nur komplette Devices einer Volume Group hinzugefügt werden können.

`pvdisplay -v /dev/sda2` zeigt die Informationen, die wir über das Physical Volume haben:

```
"/dev/sda2" is a new physical volume of "<12.00 GiB"
--- NEW Physical volume ---
PV Name           /dev/sda2
VG Name
PV Size           <12.00 GiB
Allocatable      NO
PE Size          0
Total PE         0
Free PE          0
Allocated PE     0
PV UUID          jCfo25-Qrho-5J8Z-mb4Q-3syn-6AvP-IrGpEs
```

Listing 3.18 »`pvdisplay`«

Nun erzeugen wir die Volume Group mittels `vgcreate vgreichlichplatz /dev/sda2`. Das Resultat lässt sich mithilfe von `vgdisplay -v vgreichlichplatz` überprüfen:

```
--- Volume group ---
VG Name                vgreichlichplatz
System ID
Format                 lvm2
Metadata Areas        1
Metadata Sequence No  1
VG Access              read/write
VG Status              resizable
MAX LV                 0
Cur LV                0
Open LV                0
Max PV                 0
Cur PV                1
Act PV                 1
VG Size                <12.00 GiB
PE Size                4.00 MiB
Total PE               3071
Alloc PE / Size        0 / 0
Free PE / Size         3071 / <12.00 GiB
VG UUID                Mpj6Ku-rr0v-YfDZ-Lf0u-tRdo-rBT2-zNRIZr

--- Physical volumes ---
PV Name                /dev/sda2
PV UUID                jCfo25-Qrho-5J8Z-mb4Q-3syn-6AvP-IrGpEs
PV Status              allocatable
Total PE / Free PE    3071 / 3071
```

Listing 3.19 »vgdisplay«

Die Informationen über das Physical Volume haben sich auch entsprechend verändert. Die Größe des Physical Volume wurde etwas verringert, da ein paar Verwaltungsinformationen gespeichert werden müssen. Wir sehen, dass das PV jetzt zu einer Volume Group gehört, es ist benutzbar (*allocatable*), die *Physical Extent Size* ist jetzt gesetzt, und die Größe sehen wir ebenfalls. Nun legen wir ein Volume mit dem Namen *lvhome* an. Wir haben 3071 freie *Physical Extents*, die wir auch alle nutzen wollen. Also heißt der Befehl `lvcreate -l 3071 -n lvhome vgreichlichplatz`, und wir schauen uns gleich danach das Resultat mit `lvdisplay /dev/vgreichlichplatz/lvhome` an:

```
--- Logical volume ---
LV Path                /dev/vgreichlichplatz/lvhome
LV Name                lvhome
```

```
VG Name          vgreichlichplatz
LV UUID          synBSR-4KRg-WYC7-SszF-R5I0-eS7r-SPant7
LV Write Access  read/write
LV Creation host, time centos, 2021-01-26 09:31:29 +0100
LV Status        available
# open           0
LV Size          <12.00 GiB
Current LE       3071
Segments         1
Allocation       inherit
Read ahead sectors auto
- currently set to 8192
Block-Device     253:0
```

Listing 3.20 »lvdisplay«

Als Nächstes legen wir ein Dateisystem an, der Einfachheit halber nehmen wir *ext4* (andere Dateisysteme finden Sie in Kapitel 4, »Dateisysteme«).



Die Standard-Dateisysteme der einzelnen Distributionen sind zum Zeitpunkt der Drucklegung dieser Auflage *XFS* (CentOS), *btrfs* (openSUSE) und *ext4* (Debian und Ubuntu). Es spricht selbstverständlich nichts dagegen, andere als die Standard-Dateisysteme zu verwenden.

```
mkfs -t ext4 /dev/vgreichlichplatz/lvhome
```

Listing 3.21 Dateisystem auf dem *Logical Volume* anlegen

Hiernach lässt sich das gerade erzeugte Dateisystem überall da einhängen, wo wir es haben wollen: `mount /dev/vgreichlichplatz/lvhome /home`. Um das Volume auch nach einem Reboot verfügbar zu haben, muss es noch in der */etc/fstab* eingetragen werden:

```
/dev/vgreichlichplatz/lvhome /home ext4 errors=remount-ro 0 1
```

Listing 3.22 Eintrag in der »/etc/fstab«

Damit haben wir den ersten Teil abgeschlossen: Wir haben ein Physical Volume erzeugt, dieses benutzt, um eine neue Volume Group zu erzeugen, und anschließend haben wir ein neues Volume erzeugt und in den Verzeichnisbaum eingefügt.

Folgende Kommandos wurden in diesem Abschnitt behandelt:

- ▶ `pvcreate /dev/PLATTE`
- ▶ `vgcreate VGNAME /dev/PLATTE`

- ▶ `lvcreate -L <GROESSE> -n LVNAME VGNAME`
- ▶ `mkfs -t <DATEISYSTEM> /dev/VGNAME/LVNAME`
- ▶ `mount /dev/VGNAME/LVNAME MOUNTPUNKT`
- ▶ Eintrag in */etc/fstab*
- ▶ `pvdisk /dev/PLATTE`
- ▶ `lvdisplay /dev/VGNAME/LVNAME`
- ▶ `vgdisplay VGNAME`

3.2.4 Erweiterung eines Volumes

Die Erweiterung eines Volumes erfolgt in zwei Schritten: Zum einen muss das zugrunde liegende Logical Volume vergrößert werden, und anschließend muss das enthaltene Filesystem angepasst werden.

`lvextend -L +500M /dev/vgreichlichplatz/lvhome` vergrößert das Logical Volume um 500 MB, wenn in der Volume Group der nötige Platz vorhanden ist. Allerdings ist davon im Filesystem nichts sichtbar, da es zusätzlich noch vergrößert werden muss.

Achtung bei der Verkleinerung eines Dateisystems



Auch wenn sich Dateisysteme mittlerweile verkleinern lassen, sieht die sicherste und empfohlene Methode anders aus: Legen Sie zunächst ein Backup der Daten an. Löschen Sie dann das Dateisystem, und legen Sie es wieder neu an. Spielen Sie nun die Daten aus dem Backup wieder ein (Recovery).

Im Fall von *ext4* und einem Kernel 2.6 oder neuer lässt sich das Filesystem online vergrößern. `resize2fs /dev/vgreichlichplatz/lvhome` vergrößert das Dateisystem online auf die maximal zulässige Größe, wie Sie leicht mit `df -h` überprüfen können.

Beschriebene Befehle

Folgende Kommandos wurden in diesem Abschnitt behandelt:

- ▶ `lvextend -L +<VERGROESSERUNG> /dev/VGNAME/LVNAME`
- ▶ `resize2fs /dev/VGNAME/LVNAME`

Damit ist es möglich – genügend Platz in der Volume Group vorausgesetzt –, Logical Volumes nachträglich zu vergrößern.

3.2.5 Eine Volume Group erweitern

Analog zur Erweiterung eines Volumes mit `lvextend` heißt der Befehl bei Volume Groups `vgextend`. Der Befehl `pvcreate /dev/sdb` erzeugt ein neues Physical Volume aus der zweiten Festplatte, und `vgextend vgreichlichplatz /dev/sdb` – Sie müssen dafür keine Partition erstellen, wenn Sie das Block-Device komplett verwenden wollen – fügt diese Platte dann zur Volume Group hinzu. `vgdisplay vgreichlichplatz` gibt die Information zur Volume Group raus, und `vgdisplay -v vgreichlichplatz` liefert zusätzlich die Informationen zu allen enthaltenen Objekten:

```
--- Volume group ---
VG Name                vgreichlichplatz
System ID
Format                 lvm2
Metadata Areas        2
Metadata Sequence No  3
VG Access              read/write
VG Status              resizable
MAX LV                 0
Cur LV                1
Open LV               1
Max PV                0
Cur PV                2
Act PV                2
VG Size                19.99 GiB
PE Size                4.00 MiB
Total PE              5118
Alloc PE / Size       3071 / <12.00 GiB
Free PE / Size        2047 / <8.00 GiB
VG UUID                Mpj6Ku-rr0v-YfDZ-Lf0u-tRdo-rBT2-zNRIZr

--- Logical volume ---
LV Path                /dev/vgreichlichplatz/lvhome
LV Name                lvhome
VG Name                vgreichlichplatz
LV UUID                synBSR-4KRg-WYC7-SszF-R5IO-eS7r-SPant7
LV Write Access       read/write
LV Creation host, time centos, 2021-01-26 09:31:29 +0100
LV Status              available
# open                 1
LV Size                <12.00 GiB
Current LE             3071
Segments               1
Allocation             inherit
```

```

Read ahead sectors      auto
- currently set to     8192
Block-Device           253:0

--- Physical volumes ---
PV Name                 /dev/sda2
PV UUID                 jCfo25-Qrho-5J8Z-mb4Q-3syn-6AvP-IrGpEs
PV Status               allocatable
Total PE / Free PE     3071 / 0

PV Name                 /dev/sdb
PV UUID                 rVAPEk-LKdt-M13f-I46G-IUYG-mTAs-ZMU3F9
PV Status               allocatable
Total PE / Free PE     2047 / 2047

```

Listing 3.23 »vgdisplay« mit neuem Device

Folgende Kommandos wurden in diesem Abschnitt behandelt:

- ▶ `vgextend VGNAME PVNAME`
- ▶ `vgdisplay -v VGNAME`

Eine bestehende Volume Group kann nun mit zusätzlichen Disks vergrößert werden.

3.2.6 Spiegelung zu einem Volume hinzufügen

LVM können Sie auch benutzen, um Volumes zu spiegeln. Allerdings müssen Sie beachten, dass bei Vergrößerungen der Platz auf allen benutzten Physical Volumes vorhanden sein muss. Der Befehl `lvconvert --type mirror -m 1 --mirrorlog core /dev/vgreichlichplatz/lvhome` fügt unserem *lvhome*-Volume einen Spiegel hinzu. Das ist sehr zeitintensiv. Dieser Spiegel muss zwangsweise auf einem zweiten Physical Volume liegen, sonst wäre er kein echter Spiegel. Der Parameter `core` von `mirrorlog` besagt, dass der Spiegel bei jedem Neustart wieder neu aufgebaut wird.

Es ist ebenfalls möglich, LVM auf einem Hardware- oder Software-RAID zu betreiben. Wie ein solches RAID aufgebaut wird, ist in Abschnitt 3.1, »RAID«, beschrieben.



Die knappe Eingabe `lvs` zeigt alle Daten zur Spiegelung an. Zur Komplettierung der Informationen folgen hier auch noch die Ausgaben von `vgs` und `pvs`:

```

testserver:~# pvs
PV          VG          Fmt Attr PSize  PFree
/dev/sda2  vgreichlichplatz lvm2 a-- <12.00g  0
/dev/sdb   vgreichlichplatz lvm2 a-- <20.00g  8.00g

```

```

testserver:~# vgs
VG                #PV #LV #SN Attr   VSize  VFree
vgreichlichplatz  2   1  0 wz--n- 31.99g 8.00g

testserver:~# lvs
LV      VG                Attr      LSize  [...] Meta%  Move Log Cpy%Sync Convert
lvhome  vgreichlichplatz  mwi-aom--- <12.00g [...]          100.00

```

Listing 3.24 »pvs«, »vgs« und »lvs«

Wenn Sie den Neuaufbau des Spiegels bei jedem Neustart verhindern möchten, können Sie den Parameter weglassen. Das wäre gleichbedeutend mit `mirrorlog disk`, aber das bedingt, dass es ein drittes Physical Volume in der Volume Group gibt, auf dem Sie die Spiegelungslogs ablegen können. Eine Spiegelung können Sie – wenn eine ausreichende Anzahl von Physical Volumes vorhanden ist – auch direkt bei der Erstellung einrichten. Wie das geht, wird in Abschnitt 3.2.9, »Mirroring ausführlich«, beschrieben. Es kann auch mehr als einmal gespiegelt werden: Der Parameter `-m 2` sorgt dafür, dass zwei Spiegelungen angelegt werden.



An dem Dateisystem, das auf dem Logical Volume liegt, muss nichts geändert werden. Das Verhalten des Logical Volumes ist transparent.

Folgende Kommandos wurden in diesem Abschnitt behandelt:

- ▶ `lvconvert --type mirror -m x --mirrorlog y /dev/VGNAME/LVNAME`
- ▶ `lvcreate -L GROESSE -n LVNAME -m 1 --mirrorlog y VGNAME`
- ▶ `pvs`
- ▶ `vgs`
- ▶ `lvs`

Mit den beiden ersten Befehlen können Sie Spiegelungen mittels LVM durchführen. Die neuen dreibuchstabigen Befehle helfen Ihnen zudem, einen schnellen Überblick zu bekommen.

3.2.7 Eine defekte Festplatte ersetzen

Wenn es erforderlich ist, eine Festplatte zu wechseln, sollten Sie wie folgt vorgehen: Geben Sie dem System eine weitere Festplatte (das darf auch eine USB-Festplatte sein), und verschieben Sie alle belegten Extents vom defekten Physical Volume auf die neue Festplatte. Entfernen Sie anschließend die defekte Platte aus der Volume Group, und bauen Sie die defekte Festplatte aus. Mit dem Kommando `pvmove /dev/sdb1 /dev/sda2` verschieben Sie alle belegten Extents der defekten Platte `/dev/sdb1` auf die intakte Festplatte `/dev/sda2`. Das Komman-

do `vgreduce vgreichlichplatz /dev/sdb1` entfernt die defekte Platte aus der Volume Group. Nach diesem Schritt können Sie die Platte ausbauen.

Folgende Kommandos wurden in diesem Abschnitt behandelt:

- ▶ `pvmove PV1 PV2`
- ▶ `vgreduce VGNAME PV`

Damit können Sie ohne Datenverluste eine Disk einer Volume Group entfernen.

3.2.8 Backups mit Snapshots

Snapshots sind eine gute Möglichkeit, Backups eines Systems im laufenden Betrieb zu erstellen. Ein *Snapshot* («Schnappschuss») erstellt eine Sicht auf ein Logical Volume zu einem bestimmten Zeitpunkt. Dieses Verfahren wird sehr häufig bei Datenbanken angewendet. Man hält den Datenbankserver an, macht einen Snapshot und lässt den Server danach weiterlaufen. Von dem Snapshot kann man danach eine Sicherung machen, ohne den laufenden Betrieb zu unterbrechen. Die Technik, die LVM dabei verwendet, nennt man *Copy-on-Write* oder abgekürzt COW. In dem Moment, in dem der Schnappschuss angelegt wird, überwacht der Logical Volume Manager das Ursprungs-Volume auf Änderungen und schreibt im Moment der Änderung eines Logical Extents die unveränderte Originalversion in den Snapshot.

Wir erstellen nun eine neue Volume Group namens `vg_adminbuch` und darin ein Logical Volume mit dem Namen `lv_daten`:

```
testserver:~# pvcreate /dev/sdb
Physical volume "/dev/sdb" successfully created.

testserver:~# vgcreate vg_adminbuch /dev/sdb
Volume group "vg_adminbuch" successfully created

testserver:~# lvcreate --name lv_daten -L 100M vg_adminbuch
Logical volume "lv_daten" created.

testserver:~# mkfs -t ext4 /dev/vg_adminbuch/lv_daten
mke2fs 1.45.6 (20-Mar-2020)
Creating filesystem with 102400 1k blocks and 25688 inodes
Filesystem UUID: 74cc5894-aed8-4777-9ae5-5c7e680d86fa
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
```

```
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

```
testserver:~# mount /dev/vg_adminbuch/lv_daten /mnt
```

Listing 3.25 Eine neue Volume Group mit einem neuen Logical Volume

Um ein paar Daten im Logical Volume zu haben, schreiben wir einfach den Inhalt des Systemlogs von *systemd* in das neu erstellte Volume: `journalctl > /mnt/journal`. Um einen Kontrollpunkt für unsere Bemühungen zu haben, hängen wir mit dem folgenden Befehl `echo "*** LVM- TEST ***">> /mnt/journal` noch eine aussagekräftige Nachricht an das Ende der Datei.

Nun erstellen wir einen Snapshot und prüfen, ob die letzte Zeile der Datei im Original und im Schnappschuss übereinstimmt:

```
testserver:~# lvcreate -L 4M --snapshot --name lv_snapshot /dev/vg_adminbuch/lv_daten
Logical volume "lv_snapshot" created.
```

```
testserver:~# mkdir /root/snapshot
testserver:~# mount /dev/vg_adminbuch/lv_snapshot /root/snapshot
testserver:~# tail -n 1 /mnt/journal /root/snapshot/journal
==> /mnt/journal <==
*** LVM- TEST ***
```

```
==> /root/snapshot/journal <==
*** LVM- TEST ***
```

Listing 3.26 Erstellen eines Snapshots

Bei der Erstellung des Snapshots müssen Sie angeben, von welchem ursprünglichen Volume der Schnappschuss erstellt werden soll und wie viel Platz für die Ursprungsdaten vorgehalten werden soll. Erfahrungsgemäß sollten Sie zwischen 20 und 30 % der Ursprungsgröße vorhalten. Auf »Nummer sicher« gehen Sie, wenn Sie 100 % zusätzlichen Platz bereitstellen.



Die für den Snapshot tatsächlich benötigte Größe ist immer davon abhängig, wie viele Daten sich wirklich ändern.

Die Informationen, die das Kommando `lvdisplay` über ein Snapshot-Volume zur Verfügung stellt, zeigt Ihnen Listing 3.27:

```
testserver:~# lvdisplay -v /dev/vg_adminbuch/lv_snapshot
--- Logical volume ---
LV Path                /dev/vg_adminbuch/lv_snapshot
LV Name                 lv_snapshot
VG Name                 vg_adminbuch
```



```

LV UUID                BL5ZCS-iLOH-SIme-Zl97-Yo6b-GIHK-J2KZUZ
LV Write Access        read/write
LV Creation host, time centos, 2021-01-26 10:23:28 +0100
LV snapshot status     active destination for lv_daten
LV Status              available
# open                 1
LV Size                100.00 MiB
Current LE             25
COW-table size         4.00 MiB
COW-table LE          1
Allocated to snapshot 0.49%
Snapshot chunk size    4.00 KiB
Segments               1
Allocation             inherit
Read ahead sectors     auto
- currently set to     8192
Block-Device           253:4

```

Listing 3.27 Ausgabe von »lvsdisplay« für ein Snapshot-Volume

Nun hängen wir mit echo "*** ORIGINAL ***">> /mnt/journal eine weitere Zeile an das Ende der Originaldatei und überprüfen den Snapshot:

```

testserver:~# ls -l /mnt/journal /root/snapshot/journal
-rw-r--r--. 1 root root 80207 Jan 26 10:27 /mnt/journal
-rw-r--r--. 1 root root 80190 Jan 26 10:23 /root/snapshot/journal

testserver:~# tail -n 1 /mnt/journal /root/snapshot/journal
==> /mnt/journal <==
*** ORIGINAL ***

==> /root/snapshot/journal <==
*** LVM- TEST ***

```

Listing 3.28 Überprüfen des Snapshots

Es hat also funktioniert. Die Originaldatei wurde verändert, und die Datei im Snapshot hat noch den alten Inhalt. Mit dem Befehl `lvremove /dev/vg_adminbuch/lv_snapshot` sollten Sie nach Abschluss des Backups den Schnappschuss wieder löschen, da einiges an Performance gebraucht wird, wenn jeder Schreibvorgang an den Originaldaten einen Schreibvorgang im Snapshot auslöst. Vergessen Sie bitte nicht, vorher ein `umount /root/snapshot` auszuführen.

Schnappschüsse sind auch beschreibbar! Damit bieten sie neben der Backupfunktionalität auch ein gutes Mittel, um schnell virtuelle Kopien von Daten zu erstellen und mit diesen Kopien Programme zu testen.





Es kann selbstverständlich passieren, dass sich mehr Daten ändern, als man an Platz bei der Erstellung des Schnappschusses vorgesehen hat. In diesem Fall wird der Schnappschuss ungültig. Mit den Originaldaten kann aber wie gewohnt weitergearbeitet werden.

In diesem Beispiel schreiben wir 50 MB an Daten in das Original-Volume. Wir haben für den Snapshot aber nur 4 MB vorgesehen, daher ändert sich der Status von `active` auf `INACTIVE`:

```
testserver:~# dd if=/dev/zero of=/mnt/nullen bs=1024 count=50000
50000+0 records in
50000+0 records out
51200000 bytes (51 MB, 49 MiB) copied, 0.210782 s, 243 MB/s
```

```
testserver:~# lvdisplay -v /dev/vg_adminbuch/lv_snapshot
--- Logical volume ---
LV Path                /dev/vg_adminbuch/lv_snapshot
LV Name                 lv_snapshot
VG Name                 vg_adminbuch
LV UUID                 zDhREP-3VCx-76sF-c5Ig-PdZJ-17HH-Hwyy9t
LV Write Access         read/write
LV Creation host, time centos, 2021-01-26 10:33:40 +0100
LV snapshot status      INACTIVE destination for lv_daten
LV Status               available
# open                  0
LV Size                 100.00 MiB
Current LE              25
COW-table size          4.00 MiB
COW-table LE           1
Snapshot chunk size     4.00 KiB
Segments                1
Allocation              inherit
Read ahead sectors      auto
- currently set to     8192
Block-Device            253:4
```

Listing 3.29 Überlauf des Snapshot-Volumes

Wir beenden das Beispiel, indem wir den Schnappschuss aushängen und ihn anschließend löschen:

```
testserver:~# lvremove /dev/vg_adminbuch/lv_snapshot
Do you really want to remove active logical volume vg_adminbuch/lv_snapshot? [y/n]: y
Logical volume "lv_snapshot" successfully removed
```

Listing 3.30 Das Schnappschuss-Volume entfernen

Das folgende Kommando wurde in diesem Abschnitt behandelt:

```
▶ lvcreate -L GROESSE --snapshot --name LVNAME ORIGINALLV
```

Mit diesem Kommando können Sie einen Snapshot erstellen und mit der Kopie weiterarbeiten, ohne die Originaldaten zu verändern.

3.2.9 Mirroring ausführlich

LVM ist ebenfalls in der Lage, Ihre Daten zu spiegeln. Dazu schreibt LVM die Daten auf zwei verschiedene Physical Volumes und stellt damit sicher, dass die Daten auch dann verfügbar bleiben, falls eines der Physical Volumes ausfällt. Um den Status der Spiegelung festzustellen, nutzt LVM eine Log-Datei, in der der Status der Schreibvorgänge festgehalten wird. Diese Datei wird *Mirrorlog* genannt. Für das Mirrorlog wird ein wenig Speicher im Verwaltungsbereich der Devices benötigt. Dieser Speicherplatz steht nicht für Daten zur Verfügung. Es gibt drei verschiedene konfigurierbare Methoden, um das Mirrorlog anlegen zu lassen:

1. disk

Das ist der Standard. Damit wird das Mirrorlog auf eine Disk geschrieben. Idealerweise wird es auf ein Physical Volume geschrieben, das nicht im Spiegel verwendet wird. Wenn dort aber kein Platz mehr ist, wird es auf ein Volume geschrieben, das auch eine Seite des Spiegels enthält.

2. core

Damit wird das Mirrorlog im Speicher angelegt. Bei jeder Aktivierung des Logical Volumes werden alle Daten vom ersten Physical Volume des Spiegels auf das zweite Volume kopiert, um sicherzustellen, dass der Spiegel komplett ist. Davon wird abgeraten, und diese Option sollte nur in Notfällen – gegebenenfalls temporär – verwendet werden.

3. mirrored

Das ist der empfohlene Weg: Das Mirrorlog wird in diesem Fall ebenfalls gespiegelt, sodass im Fehlerfall eine Spiegelung schnell wieder aufgebaut werden kann.

Spiegel können über mehr als zwei Physical Volumes verteilt werden, sodass der Ausfall eines einzigen Device nicht zum Ausfall des Spiegels führt.

Für die Beispiele hier im Buch werden zwei Physical Volumes mit jeweils 20 GB Größe verwendet. Diese werden zum Aufbau einer neuen Volume Group *vg_mirror* verwendet (eine neue Volume Group wäre nicht nötig, sie dient hier nur der Verdeutlichung):

```
testserver:~# pvs /dev/sdb /dev/sdc
PV          VG          Fmt Attr PSize  PFree
/dev/sdb    vg_mirror   lvm2 a--  <20.00g <20.00g
/dev/sdc    vg_mirror   lvm2 a--  <20.00g <20.00g
```

```
testserver:~# vgs vg_mirror
VG          #PV #LV #SN Attr   VSize VFree
vg_mirror   2   0   0 wz--n- 39.99g 39.99g
```

Listing 3.31 Ausgangssituation

Die einfachste Form, ein gespiegeltes Logical Volume zu erzeugen, besteht darin, zusätzlich den Parameter `-m 1` beim Erstellen zu verwenden:

```
testserver:~# lvcreate -L 5G -m 1 -n lv_spiegel vg_mirror /dev/sdb /dev/sdc
Logical volume "lv_spiegel" created.
testserver:~# lvs --all vg_mirror
LV          VG          Attr      LSize Pool [...] Cpy%Sync Convert
lv_spiegel  vg_mirror  rwi-a-r--- 5.00g  [...] 100.00
[lv_spiegel_rimage_0] vg_mirror  iwi-a-or--- 5.00g
[lv_spiegel_rimage_1] vg_mirror  iwi-a-or--- 5.00g
[lv_spiegel_rmeta_0]  vg_mirror  ewi-a-or--- 4.00m
[lv_spiegel_rmeta_1]  vg_mirror  ewi-a-or--- 4.00m
```

Listing 3.32 Erzeugen eines gespiegelten Logical Volumes

Es ist bei der Ausgabe des Befehls `lvs` in Listing 3.32 deutlich zu sehen, aus welchen Teilen das gespiegelte Volume besteht (Spiegelteil 1, 2 und die Metadaten), und die Ausgabe von `pvs` zeigt durch den freien Speicherplatz, dass die Log-Datei nur auf einer Seite angelegt wurde. Der Spiegel ist zu 100 % synchron. Bei Spiegeln, die aus großen Teilen bestehen, lohnt es sich, den Parameter `--nosync` mitzugeben. Das sorgt dafür, dass initial nicht der Inhalt des ersten Teils mit dem zweiten (oder weiteren) Teil(en) des Spiegels synchronisiert wird.



In früheren Versionen von LVM war der Standardtyp »mirror«. Der neue Standard bei gespiegelten Logical Volumes ist »RAID1«.

In unserem Beispiel in Listing 3.32 wurden die Physical Volumes für die Spiegelung explizit angegeben. Bei Volume Groups, die nur aus zwei Volumes bestehen, wäre das nicht nötig. Wenn Sie aber mehr als zwei Volumes haben, können Sie darüber selbst steuern, wo der Spiegel aufgebaut wird:

```
testserver:~# pvdisplay --maps /dev/sdb /dev/sdc
--- Physical volume ---
PV Name           /dev/sdb
VG Name           vg_mirror
PV Size           20.00 GiB / not usable 4.00 MiB
Allocatable       yes
PE Size           4.00 MiB
Total PE          5119
Free PE           3838
Allocated PE      1281
```

```

PV UUID                AKGhty-eYSv-aT8A-zYjZ-04gn-nVD5-1LHzHX

--- Physical Segments ---
Physical extent 0 to 0:
  Logical volume      /dev/vg_mirror/lv_spiegel_rmeta_0
  Logical extents     0 to 0
Physical extent 1 to 1280:
  Logical volume      /dev/vg_mirror/lv_spiegel_rimage_0
  Logical extents     0 to 1279
Physical extent 1281 to 5118:
  FREE

--- Physical volume ---
PV Name                /dev/sdc
VG Name                vg_mirror
PV Size                20.00 GiB / not usable 4.00 MiB
Allocatable           yes
PE Size                4.00 MiB
Total PE               5119
Free PE                3838
Allocated PE           1281
PV UUID                nLbVm1-to4u-dVA7-LZ09-FUhj-t51A-6AsEAj

--- Physical Segments ---
Physical extent 0 to 0:
  Logical volume      /dev/vg_mirror/lv_spiegel_rmeta_1
  Logical extents     0 to 0
Physical extent 1 to 1280:
  Logical volume      /dev/vg_mirror/lv_spiegel_rimage_1
  Logical extents     0 to 1279
Physical extent 1281 to 5118:
  FREE

```

Listing 3.33 Verteilung des Spiegels über die beteiligten Physical Volumes

Die Ausgabe in Listing 3.33 zeigt Ihnen, wie die Verteilung der Daten des Logical Volumes auf die Physical Volumes aussieht und wie viel freier Platz noch vorhanden ist.

Ein solches Logical Volume können Sie wie jedes andere Logical Volume verwenden, also es so formatieren und einbinden, wie Sie es aus früheren Beispielen kennen.

Ausfall einer Festplatte in einem Mirror

Wenn eine Festplatte aus diesem Verbund wegfällt, kann die Volume Group nicht mehr automatisch aktiviert werden.

Wir müssen die Volumes explizit aktivieren, um zu zeigen, dass wir das auch wollen:

```
testserver:~# vgs
WARNING: Couldn't find device with uuid nLbVm1-[...]-6AsEAj.
WARNING: VG vg_mirror is missing PV nLbVm1-[...]-6AsEAj (last written to /dev/sdc).
VG          #PV #LV #SN Attr   VSize  VFree
vg_mirror   2   1   0 wz-pn- 39.99g 29.98g
testserver:~# ls -ld /dev/vg_mirror
ls: cannot access '/dev/vg_mirror': No such file or directory
```

Listing 3.34 Die Situation bei einem weggefallenen Device

Die Vorgehensweise ist, das Physical Volume aus der Volume Group zu entfernen, um danach die Volume Group aktivieren und wieder auf die Daten zugreifen zu können:

```
testserver:~# vgreduce --removemissing --force vg_mirror
WARNING: Couldn't find device with uuid nLbVm1-[...]-6AsEAj.
WARNING: VG vg_mirror is missing PV nLbVm1-[...]-6AsEAj (last written to /dev/sdc).
WARNING: Couldn't find device with uuid nLbVm1-[...]-6AsEAj.
WARNING: Couldn't find device with uuid nLbVm1-[...]-6AsEAj.
Wrote out consistent volume group vg_mirror.
testserver:~# vgs
VG          #PV #LV #SN Attr   VSize  VFree
vg_mirror   1   1   0 wz--n- <20.00g 14.99g
testserver:~# vgchange -ay --partial vg_mirror
PARTIAL MODE. Incomplete logical volumes will be processed.
1 logical volume(s) in volume group "vg_mirror" now active
testserver:~# mount /dev/vg_mirror/lv_spiegel /mnt
```

Listing 3.35 Defektes Physical Volume entfernen

Der letzte Befehl mountet das jetzt nicht mehr gespiegelte Logical Volume. Wenn wir jetzt eine neue Festplatte oder allgemeiner gesagt ein neues Device haben, so können wir dieses der Volume Group hinzufügen und eine neue Spiegelung aufsetzen:

```
testserver:~# pvcreate /dev/sdc
Physical volume "/dev/sdc" successfully created.
testserver:~# vgextend vg_mirror /dev/sdc
Volume group "vg_mirror" successfully extended
testserver:~# lvconvert --repair /dev/vg_mirror/lv_spiegel
Attempt to replace failed RAID images (requires full device resync)? [y/n]: y
Faulty devices in vg_mirror/lv_spiegel successfully replaced.
```

Listing 3.36 Neues Physical Volume zur Spiegelung verwenden



Achtung! Die Spiegelung muss für jedes betroffene Logical Volume neu aufgesetzt werden.

3.2.10 Thin Provisioning

Mit *Thin Provisioning* bezeichnet man die Bereitstellung von Speicherplatz, der häufig im virtuellen Umfeld zu finden ist. Anders als bei der normalen Provisionierung wird nicht sofort der komplette Speicherplatz zur Verfügung gestellt, sondern es wird dem System »vorgegaukelt«, dass der Speicher vorhanden ist. Es wird jedoch nur der Speicher benutzt, der auch tatsächlich belegt wird. Der Hintergrund dieses Verfahrens ist eine Mischkalkulation, zum Beispiel bei Heimatverzeichnissen: Wenn Sie jedem Nutzer 5 GB zugestehen, wird es immer solche geben, die nur einige wenige MB nutzen, und andere, die den Speicherplatz ausschöpfen. Thin Provisioning sorgt dafür, dass nur der Speicherplatz vergeben wird, der auch tatsächlich genutzt wird.

Aber Achtung! Sollten Sie weniger Gesamtspeicherplatz haben als Speicher, den Sie zur Verfügung stellen, spricht man von *Überprovisionierung* oder *Over-Provisioning*. Das ist der Normalfall und der häufigste Anwendungszweck für Thin Provisioning. In diesem Fall müssen Sie den wirklich verwendeten Speicherplatz überwachen! Ein zweiter Grund dafür, *Thin Provisioning* einzusetzen, besteht in der Bereitstellungsgeschwindigkeit. Der Speicher kann sofort zugewiesen werden, und die Volume Group muss erst dann erweitert werden, wenn es notwendig wird.



Begriffe

Im Umfeld von Thin Provisioning innerhalb von Logical Volume Groups gibt es Begriffe, die immer wieder auftauchen. Das Verständnis dieser Begriffe ist elementar, um sicher mit dem Thema umgehen zu können:

- ▶ **ThinDataLV**
In diesem Logical Volume werden die Daten von zur Verfügung gestellten »thin provisioned« Logical Volumes – *ThinLV* – verwaltet.
- ▶ **ThinMetaLV**
enthält die Zuordnungen der Blöcke aus dem ThinDataLV zu ThinLVs.
- ▶ **ThinPoolLV**
besteht aus einem ThinDataLV und einem ThinMetaLV (Basis für ThinLVs).
- ▶ **ThinLV**
ist das eigentliche Volume zur Benutzung durch das System. Es ist am Anfang leer und wird bei Benutzung vergrößert.
- ▶ **SnapLV**
sind Logical Volumes, die Snapshots von ThinLVs enthalten.

Vorgehensweise

Zunächst erstellen wir in der Volume Group Daten die beiden Bestandteile eines ThinPools, nämlich das Logical Volume für die Daten und das Pendant für die Metadaten.

Die nötigen Befehle werden in Listing 3.37 gezeigt:

```
testserver:~# lvcreate -n ThinDataLV -L 3G daten
  Logical volume "ThinDataLV" created.
testserver:~# lvcreate -n ThinMetaLV -L 52M daten
  Logical volume "ThinMetaLV" created.
testserver:~# lvs daten
  LV          VG   Attr      LSize Pool Origin [...]
  ThinDataLV  daten -wi-a----- 3.00g
  ThinMetaLV  daten -wi-a----- 52.00m
```

Listing 3.37 Erstellung der Bestandteile eines ThinPools

Im nächsten Schritt kombinieren wir die beiden Teile zu einem ThinPool. In Listing 3.38 sehen Sie, wie das gemacht wird. In diesem Schritt wird das bestehende ThinDataLV umbenannt, und zwar in das versteckte *ThinDataLV_tdata*. Das Gleiche passiert mit ThinMetaLV, das umbenannt wird zu *Thin-DataLV_tmeta*. Das ThinPoolLV bekommt den Namen des vorherigen Datenvolumens Thin-DataLV. Wie Sie in der Ausgabe von `lvs` sehen können, gibt die Prozentzahl hinter ThinDataLV an, wie viel vom Daten- und Metadaten-Volume bereits verwendet wird. Das `lvol0_pmspare` dient als »Überlaufschutz«, falls das Metadaten-Volume vollzulaufen droht.

```
testserver:~# lvconvert --type thin-pool --poolmetadata daten/ThinMetaLV \
daten/ThinDataLV
  Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.
  WARNING: Converting daten/ThinDataLV and daten/ThinMetaLV to thin pool's data \
          and metadata volumes with metadata wiping.
  THIS WILL DESTROY CONTENT OF LOGICAL VOLUME (filesystem etc.)
Do you really want to convert daten/ThinDataLV and daten/ThinMetaLV? [y/n]: y
  Converted daten/ThinDataLV and daten/ThinMetaLV to thin pool.
```

```
testserver:~# lvs --all daten
  LV          VG   Attr      LSize Pool Origin Data% Meta% [...]
  ThinDataLV  daten twi-a-tz-- 3.00g          0.00 10.09
  [ThinDataLV_tdata] daten Twi-ao----- 3.00g
  [ThinDataLV_tmeta] daten ewi-ao----- 52.00m
  [lvol0_pmspare]   daten ewi----- 52.00m
```

Listing 3.38 Erstellung des ThinPools

Viel mehr ist schon gar nicht mehr zu tun, außer dass wir unser erstes ThinLV anlegen wollen. Listing 3.39 zeigt Ihnen, wie das geht:

```
testserver:~# lvcreate -n gigabyte -V 1G --thinpool daten/ThinDataLV
  Logical volume "gigabyte" created.
```



```
testserver:~# lvcreate -n terrabyte -V 1T --thinpool daten/ThinDataLV
WARNING: Sum of all thin volume sizes (1.00 TiB) exceeds the size of thin pool \
        daten/ThinDataLV and the size of whole volume group (39.99 GiB).
WARNING: You have not turned on protection against thin pools running out of space.
WARNING: Set activation/thin_pool_autoextend_threshold below 100 to trigger \
        automatic extension of thin pools before they get full.
Logical volume "terrabyte" created.
```

```
testserver:~# lvs daten
LV          VG   Attr      LSize Pool      Origin Data%  Meta% [...]
ThinDataLV daten twi-aotz-- 3.00g
gigabyte   daten Vwi-a-tz-- 1.00g ThinDataLV 0.00
terrabyte  daten Vwi-a-tz-- 1.00t ThinDataLV 0.00
```

Listing 3.39 Erstellung von ThinLVs

Beachten Sie bitte auch die Warnmeldung: Wir haben mehr Speicherplatz zugewiesen, als verfügbar ist. In den folgenden Listings sind die Warnungen wegen Überprovisionierung ausgeblendet. Eine Besonderheit von Snapshots im Thin-Provisioning-Umfeld ist, dass den Snapshots kein Speicher zugewiesen werden muss und dass sie separat aktiviert werden. Das zeigt das kleine »k« in der Tabelle von `lvs daten` aus Listing 3.40. Das fehlende »a« und die fehlende Zahl in der Spalte Data deuten darauf hin, dass ein Volume noch nicht aktiviert ist.



```
testserver:~# lvcreate -n gb_snap -s daten/gigabyte
[...]
Logical volume "gb_snap" created.
```

```
testserver:~# lvcreate -n tb_snap -s daten/terrabyte
[...]
Logical volume "tb_snap" created.
```

```
testserver:~# lvchange -ay -K daten/tb_snap
testserver:~# lvs daten
LV          VG   Attr      LSize Pool      Origin  Data%  Meta% [...]
ThinDataLV daten twi-aotz-- 3.00g
gb_snap    daten Vwi---tz-k 1.00g ThinDataLV gigabyte
gigabyte   daten Vwi-a-tz-- 1.00g ThinDataLV 0.00
tb_snap    daten Vwi-a-tz-k 1.00t ThinDataLV terrabyte 0.00
terrabyte  daten Vwi-a-tz-- 1.00t ThinDataLV 0.00
```

Listing 3.40 Erstellung von ThinLVs

In diesem Abschnitt haben wir keine neuen Befehle angewendet, nur neue Optionen von bereits bekannten Befehlen eingesetzt. Weitere Informationen zu Thin Provisioning finden Sie auf der Manpage `lvmthin`.

3.2.11 Kommandos

Die Bedeutung der Abkürzungen von wichtigen LVM-Begriffen finden Sie im Kasten zu Beginn von Abschnitt 3.2.1, »Grundlagen und Begriffe«. Hier folgen zusammengefasst die Befehle, die Sie bei LVM einsetzen können:

- ▶ **pvdisplay**: Attribute eines PVs anzeigen
- ▶ **vgdisplay**: Attribute einer VG anzeigen
- ▶ **lvdisplay**: Attribute eines LVs anzeigen
- ▶ **pvs**: Informationen über PVs anzeigen
- ▶ **vgs**: Informationen über VGs anzeigen
- ▶ **lvs**: Informationen über LVs anzeigen
- ▶ **pvscan**: alle Disks nach PVs durchsuchen
- ▶ **vgscan**: alle Disks nach VGs durchsuchen und Caches erneuern
- ▶ **lvscan**: alle Disks nach LVs durchsuchen
- ▶ **pvck**: Metadaten eines PVs prüfen
- ▶ **vgck**: Metadaten einer VG prüfen
- ▶ **pvcreate**: ein PV für die Nutzung mit LVM vorbereiten
- ▶ **vgcreate**: eine VG erstellen
- ▶ **lvcreate**: ein LV in einer bestehenden VG erzeugen
- ▶ **pvchange**: Attribute eines PVs ändern
- ▶ **vgchange**: Attribute einer VG ändern
- ▶ **lvchange**: Attribute eines LVs ändern
- ▶ **vgextend**: ein PV zu einer VG hinzufügen
- ▶ **lvextend**: die Größe eines LVs ändern
- ▶ **vgreduce**: eine VG durch das Entfernen von PVs verkleinern
- ▶ **lvreduce**: die Größe eines LVs ändern
- ▶ **pvresize**: ein PV in der Größe verändern bzw. anpassen
- ▶ **lvresize**: die Größe eines LVs ändern
- ▶ **pvremove**: ein PV entfernen (Löschen der LVM-Metadaten)
- ▶ **vgremove**: eine VG entfernen
- ▶ **lvremove**: ein LV entfernen
- ▶ **pvmove**: ein PE von einem PV auf ein anderes kopieren
- ▶ **vgexport**: eine VG für das System unsichtbar machen
- ▶ **vgimport**: eine exportierte VG wieder sichtbar machen

- ▶ **vgcfgbackup**: eine Sicherung der VG-Metadaten erstellen
- ▶ **vgcfgrestore**: Rücksicherung der VG-Metadaten
- ▶ **vgconvert**: Konvertieren der Metadaten (von/zu LVM1 nach/von LVM2)
- ▶ **lvconvert**: ein LV von linear nach Mirror oder Snapshot konvertieren
- ▶ **vgrename**: eine VG umbenennen
- ▶ **lvrename**: ein LV umbenennen
- ▶ **vgmerge**: zwei VGs zusammenführen
- ▶ **vgsplit**: ein VG in zwei VGs teilen, LVs von einer VG in die andere durch das Verschieben der PVs übertragen
- ▶ **vgmknodes**: ein VG-Verzeichnis und LV-*special files* neu erzeugen
- ▶ **lvmchange**: Attribute des LVMS ändern
- ▶ **lvmdiskscan**: nach allen Devices scannen, die für LVM2 sichtbar sind
- ▶ **lvmdump**: LVM2-Dumps zur Diagnose erstellen

3.3 udev

Die Kernelfunktion *udev* verwaltet nicht nur die Gerätedateien unter */dev*, sie erzeugt auch dynamisch neue, wenn diese gebraucht werden. Die sogenannten *device uevents* werden beim Booten des Systems oder beim Anschließen externer Geräte erzeugt. Der *udev*-Daemon fängt diese Prozesse ab und bearbeitet sie weiter.

Das erfolgt nach bestimmten *udev*-Regeln, die notwendig sind, damit *udev* weiß, welche Ereignisse ausgelöst werden sollen. Um die Geräte nutzen zu können, laden die *udev*-Rules benötigte Module nach oder führen Programme aus und sorgen so für einen reibungslosen Ablauf.

Mit den festen Regeln für eine dauerhafte Zuordnung von Geräten zu Gerätedateien ist dieses System von großem Vorteil – im Gegensatz zu früheren Varianten, bei denen das nicht immer möglich war.

3.3.1 udev-Regeln

Die *udev*-Rules sind, gegliedert nach ihrem Aufgabenbereich, in unterschiedliche Verzeichnisse aufgeteilt. Die Standardregeln finden Sie unter */lib/udev/rules.d*, systemspezifische oder selbst aufgestellte Regeln finden Sie in */etc/udev/rules.d*.

Die Reihenfolge, in der die Regeldateien angewendet werden, gibt ein Nummernpräfix vor. Eine Liste aller Dateien zeigt Listing 3.41. Die Dateien finden Sie im Verzeichnis der Standardregeln */lib/udev/rules.d*:

```
root@debian:~# ls /lib/udev/rules.d
40-hplip.rules                77-mm-nokia-port-types.rules
40-usb_modeswitch.rules      77-mm-pcmcia-device-blacklist.rules
42-qemu-usb.rules            77-mm-platform-serial-whitelist.rules
50-udev-default.rules        77-mm-qdl-device-blacklist.rules
55-dm.rules                  77-mm-simtech-port-types.rules
56-hpmud_support.rules       77-mm-usb-device-blacklist.rules
60-bridge-network-interface.rules 77-mm-x22x-port-types.rules
60-cdrom_id.rules            77-mm-zte-port-types.rules
60-crda.rules                77-nm-olpc-mesh.rules
60-fuse.rules                78-sound-card.rules
60-gnupg.rules               80-drivers.rules
[...]
```

Listing 3.41 Regeldateien in »/lib/udev/rules.d«

3.3.2 Eigene Regeln schreiben

In einzelnen Fällen ist es angebracht, eigene Regeln zu schreiben. Arbeiten Sie mit externen Geräten, wie zum Beispiel mit einer oder mehreren USB-Festplatten, wird der Gerätename immer in der Reihenfolge vergeben, in der Sie die USB-Festplatten angeschlossen haben. In Backupskripten beispielsweise ist unter `/dev/sdx` nicht zwangsweise immer das externe Backuplaufwerk zu finden – sinnvoll wäre es aber, wenn die externen Geräte immer mit dem gleichen Devicenamen angesprochen werden, um sie im Skript zweifelsfrei identifizieren zu können.

Es ist also notwendig, die Merkmale des Laufwerks eindeutig zu erkennen, um ihm einen dauerhaften Namen zuordnen zu können. Anhand der einmaligen Merkmale kann dann die *udev*-Regel geschrieben werden. Als Erstes ist es wichtig, nach dem Einstecken des Geräts zu prüfen, was *udev* standardmäßig damit macht. Mit dem Kommando `dmesg` können Sie wie in Listing 3.42 sehen, was genau passiert:

```
[39060.775487] usb 1-1.5.1: new high-speed USB device number 11 using ehci_hcd
[39060.868866] usb 1-1.5.1: New USB device found, idVendor=1058, idProduct=1021
[39060.868871] usb 1-1.5.1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[39060.868875] usb 1-1.5.1: Product: Ext HDD 1021
[39060.868878] usb 1-1.5.1: Manufacturer: Western Digital
[39060.868880] usb 1-1.5.1: SerialNumber: 574341563537373432363734
[39060.869534] scsi7 : usb-storage 1-1.5.1:1.0
[39061.866206] scsi 7:0:0:0: Direct-Access WD Ext HDD 1021 2002 PQ: 0 ANSI: 4
[39061.868027] sd 7:0:0:0: Attached scsi generic sg1 type 0
[39061.868187] sd 7:0:0:0: [sdb] 1953519616 512-byte logical blocks: \
(1.00 TB/931 GiB)
[39061.869512] sd 7:0:0:0: [sdb] Test WP failed, assume Write Enabled
```

```
[39061.870612] sd 7:0:0:0: [sdb] Asking for cache data failed
[39061.870620] sd 7:0:0:0: [sdb] Assuming drive cache: write through
[39061.873724] sd 7:0:0:0: [sdb] Test WP failed, assume Write Enabled
[39061.874844] sd 7:0:0:0: [sdb] Asking for cache data failed
[39061.874851] sd 7:0:0:0: [sdb] Assuming drive cache: write through
[39061.882181] sdb: sdb1
[39061.884481] sd 7:0:0:0: [sdb] Test WP failed, assume Write Enabled
[39061.885598] sd 7:0:0:0: [sdb] Asking for cache data failed
[39061.885607] sd 7:0:0:0: [sdb] Assuming drive cache: write through
[39061.885613] sd 7:0:0:0: [sdb] Attached SCSI disk
[39076.756975] EXT4-fs (sdb1): mounted filesystem with ordered data mode. Opts: \
(null)
```

Listing 3.42 Auszug aus »dmesg«

Der obige Auszug aus *dmesg* zeigt, dass *udev* die USB-Festplatte selbstständig erkennt, automatisch den Treiber lädt und die Festplatte unter */dev/sdb* zur Verfügung stellt.

Die für jedes beteiligte Gerät relevanten Informationen lesen Sie mit *udevadm* aus. Diese sehr umfangreiche Ausgabe beinhaltet nicht nur die Informationen der USB-Festplatte, ebenfalls gehören der USB-Bus, der USB-Controller, der über den PCI-Bus angebunden ist, und viele weitere Teillinformationen dazu. Ausgegeben werden die Informationen in einem sogenannten Block. Mit dem Kommando *looking at device* wird jeder einzelne Block eines sogenannten *Parents* eingeleitet.

Einen Ausschnitt sämtlicher Geräteinformationen, die mit dem Kommando *udevadm* ausgelesen wurden, sehen Sie in Listing 3.43:

```
root@debian:~# udevadm info --query=all --attribute-walk --name=/dev/sdb
[...]
  looking at device '/devices/pci0000:00/0000:00:1a.0/\
usb1/1-1/1-1.5/1-1.5.1/1-1.5.1:1.0/host7/target7:0:0/7:0:0/block/sdb':
    KERNEL=="sdb"
    SUBSYSTEM=="block"
    DRIVER==""
[...]
  looking at parent device '/devices/pci0000:00/0000:00:1a.0/\
usb1/1-1/1-1.5/1-1.5.1/1-1.5.1:1.0/host7/target7:0:0/7:0:0':
    KERNELS=="7:0:0:0"
    SUBSYSTEMS=="scsi"
    DRIVERS=="sd"
    [...]
    ATTRS{serial}=="574341563537373432363734"
[...]
```

Listing 3.43 Informationen von */dev/sdb*

Zum Definieren einer Regel können Sie die Ausgabe direkt nutzen, es dürfen aber keine Attribute von verschiedenen *Parents* vermischt werden. Die Seriennummer eines Geräts eignet sich immer gut, da sie definitiv eindeutig ist und sie sich so direkt auf die Attribute eines *Parents* und des Geräts bezieht.

Unter */etc/udev/rules.d* legen Sie nun eine eigene Datei an. In der Datei *README* des Regelverzeichnisses finden Sie Hilfe zur korrekten Nummerierung der Regel. In Listing 3.44 legen wir die Datei *71-custom-storage.rules* an und füllen sie mit folgendem Inhalt:

```
SUBSYSTEMS=="usb", KERNEL=="sd?1", ATTRS{serial}=="574341563537373432363734", \
    SYMLINK+="backup-storage"
```

Listing 3.44 Regel für */dev/sdb*

Dass es sich um ein Gerät am USB-Bus handelt, zeigt *SUBSYSTEMS*. Wie wir bereits wissen, legt *udev* für die einzelnen Gerätepartitionen unterschiedliche Gerätenamen an. Diesen Umstand nutzen wir mit der Angabe von *KERNEL=="sd?1"*. Unter */dev/sdb1* steht uns bereits die erste Partition des Geräts zur Verfügung, und genau auf diese Partition soll sich auch später der Gerätenamen beziehen. Trotzdem bleibt die Seriennummer des Geräts unser wichtigster Filter. Dafür, dass die USB-Festplatte künftig über den Namen »backup-storage« erreichbar ist, sorgt der Eintrag *SYMLINK+=*. Die erneute Ausführung der *udev*-Regeln wird durch das Kommando *udevadm trigger* forciert (siehe Listing 3.45). Dann steht Ihnen der neue Gerätenamen zur Verfügung, und Sie können ihn wie gewohnt nutzen.

```
root@debian:~# udevadm trigger
root@debian:~# mount /dev/backup-storage /mnt
root@debian:~# df -h /mnt
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb1       917G  486G  385G  56% /mnt
```

Listing 3.45 »udevadm trigger«

Wenn Sie die Konfiguration um den Eintrag *RUN+="PROGRAMMNAME"* ergänzen, können Sie zeitgleich Programme starten, die sich bereits auf das neue Gerät beziehen. Sie möchten zum Beispiel, dass direkt nach dem Einstecken der Festplatte ein Backupskript aktiviert wird? Dann definieren Sie dafür folgende Regel:

```
SUBSYSTEMS=="usb", KERNEL=="sd?1", ATTRS{serial}=="574341563537373432363734", \
    SYMLINK+="backup-storage", RUN+="/usr/bin/backintime"
```

Listing 3.46 Backup starten, sobald das Gerät angeschlossen wurde



Die Verwaltung Ihrer Geräte in Ihrem bestehenden System lässt sich also mit *udev* einfach und unkompliziert Ihren Wünschen anpassen. Allerdings führt eine falsche Konfiguration dazu, dass Ihr System unter Umständen nicht mehr startet.

3.4 Alles virtuell? »/proc«

Unter `/proc` finden Sie auf jedem Linux-System ein virtuelles *Pseudodateisystem*, in dem sich sehr viele Informationen des Linux-Kernels finden und zum Teil auch verändern lassen. So gibt es dort beispielsweise Informationen zu allen Prozessen, die das Programm `top` aufbereitet zur Verfügung stellt, oder auch Informationen über den Speicher, das Netzwerk, Treiber und vieles andere mehr. In diesem Abschnitt stellen wir Ihnen ausgewählte Informationen zur Verfügung, sodass Sie einen umfassenden Überblick über Ihr System bekommen.

3.4.1 CPU

CPU-Informationen finden Sie in der virtuellen Datei `/proc/cpuinfo`, die Sie nicht mit einem Editor bearbeiten, sondern nur anzeigen können (siehe Listing 3.47). Wie allgemein üblich, beginnt die Nummerierung bei 0.

```
root@debian:~# cat /proc/cpuinfo
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 37
model name    : Intel(R) Core(TM) i3 CPU          M 390 @ 2.67GHz
stepping      : 5
microcode     : 0x2
cpu MHz       : 933.000
cache size    : 3072 KB
physical id   : 0
siblings      : 4
core id       : 0
cpu cores     : 2
apicid        : 0
initial apicid : 0
fpu           : yes
fpu_exception : yes
cpuid level   : 11
wp            : yes
flags         : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov \
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx rdtscp lm \
constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf \
pni dtes64 monitor ds_cpl vmx est tm2 ssse3 cx16 xtpr pdcm pcid sse4_1 sse4_2 \
popcnt lahf_lm arat dtherm tpr_shadow vnmi flexpriority ept vpid
bogomips     : 5319.90
clflush size  : 64
```

```
cache_alignment : 64
address sizes   : 36 bits physical, 48 bits virtual
power management:
```

```
ssse3 cx16 sse4_1 lahf_lm
[...]
```

Listing 3.47 CPU-Informationen

Vermutlich wussten Sie schon, welche CPU Ihr System verwendet, aber in der Ausgabe finden Sie darüber hinaus die Taktfrequenz, mit der der Prozessor momentan läuft, und, was besonders wichtig ist, welche Funktionen die CPU unterstützt.

Diese Funktionen werden unter `flags` zusammengefasst. Interessant sind dabei insbesondere die folgenden ausgewählten Features:

- ▶ **lm**
steht für *long mode* und zeigt, dass wir eine 64-Bit-CPU vor uns haben.
- ▶ **ht**
steht für *hyperthreading* und gibt an, dass dieses Feature vom Prozessor unterstützt wird.
- ▶ **svm**
(nicht in unserem Beispiel vorhanden) gibt das Vorhandensein der *Secure Virtual Machine* an, die die Hardwarevirtualisierung von AMD ist.
- ▶ **vmx**
zeigt, dass der Prozessor die Hardwarevirtualisierung VMX von Intel unterstützt.



Die Ausgabe zeigt nur, welche Funktionen von der CPU insgesamt unterstützt werden, aber nicht, ob sie auch im BIOS des Computers eingeschaltet wurden.

3.4.2 RAM

Analog zur CPU-Information `/proc/cpuinfo` lassen sich Informationen zur Speichernutzung unter `/proc/meminfo` finden:

```
root@debian:~# cat /proc/meminfo
MemTotal:      7991496 kB
MemFree:       216068 kB
Buffers:       246168 kB
Cached:        4833504 kB
SwapCached:    128 kB
Active:        4519088 kB
Inactive:      2990956 kB
...
```

Listing 3.48 Speicherinformationen



Insgesamt ist die Auflistung rund 50 Zeilen lang. Linux nutzt allen nicht gebrauchten Speicher für `Buffers` und `Cached`, dementsprechend niedrig ist der Wert für den freien Speicher `MemFree`. Wenn Sie aber die Werte von `Buffers` und `Cached` hinzuzählen, kommen Sie in diesem Fall auf über 5 GB verwendbaren Speicher.

Ein besonderes Augenmerk sollten Sie bei 32-Bit-Systemen auf die Werte von `HighFree` und `LowFree` richten. Da der Kernel für seine Strukturen nur den `Low-Memory`-Bereich verwendet, können Sie eine »Speicher voll«-Meldung bekommen, obwohl noch ausreichend freier Platz im `High-Memory`-Bereich verfügbar ist. Listing 3.49 zeigt Ihnen eine Beispielausgabe eines 32-Bit-Systems:

```
root@debian:~# cat /proc/meminfo
[...]
HighTotal:      1179520 kB
HighFree:       55688 kB
LowTotal:       897368 kB
LowFree:        322108 kB
[...]
```

Listing 3.49 Speicherauslastung eines 32-Bit-Systems

Auch wenn es kaum noch 32-Bit-Systeme für den Serverbereich zu kaufen gibt, installieren immer noch einige Administratoren 32-Bit-Betriebssysteme auf 64-Bit-Systemen.

Irrglaube

Oft hört man, dass sich die Umstellung auf 64 Bit erst bei Speicher jenseits der 4 GB lohne. Fakt ist aber, dass die Begrenzung von `Low-` und `High-Memory` schon vorher stattfindet. Zusätzlich verarbeitet ein 64-Bit-System pro Prozessorakt die doppelte Menge an Daten, wenn es die verwendeten Programme erlauben.



3.4.3 Kernelkonfiguration

Von den im Buch verwendeten Distributionen bietet nur openSUSE eine virtuelle Datei namens `/proc/config.gz`, die die aktuelle Kernelkonfiguration enthält. Einen kurzen Ausschnitt aus der Datei sehen Sie in Listing 3.50. Mit diesen Informationen können Sie prüfen, mit welchen Optionen der laufende Kernel übersetzt wurde:

```
#
# Automatically generated file; DO NOT EDIT.
# Linux/x86_64 3.11.10 Kernel Configuration
#
CONFIG_64BIT=y
CONFIG_X86_64=y
```

```
CONFIG_X86=y
CONFIG_INSTRUCTION_DECODER=y
CONFIG_OUTPUT_FORMAT="elf64-x86-64"
CONFIG_ARCH_DEFCONFIG="arch/x86/configs/x86_64_defconfig"
CONFIG_LOCKDEP_SUPPORT=y
CONFIG_STACKTRACE_SUPPORT=y
[...]
```

Listing 3.50 Auszug aus der Datei »config.gz«

3.4.4 Kernelparameter

Die Parameter, mit denen der aktuell laufende Kernel gestartet wurde, finden Sie unter */proc/cmdline*:

```
centos# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-3.10.0-327.22.2.el7.x86_64 root=/dev/mapper/centos_centos-root \
ro crashkernel=auto rd.lvm.lv=centos_centos/root rd.lvm.lv=centos_centos/swap rhgb \
quiet LANG=en_US.UTF-8
```

```
opensuse# cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinuz-4.1.27-27-default root=UUID=0f4f79aa-7544-44b0-a8b7-\
d1f1947cd24f resume=/dev/disk/by-uuid/54c1a2e0-c4d6-4850-b639-7a5af8ef4339 \
splash=silent quiet showopts
```

```
debian# cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinuz-3.16.0-4-amd64 root=UUID=eac6da17-314e-43c0-956f-\
379457a505fa ro quiet
```

```
ubuntu:# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-4.4.0-31-generic root=/dev/mapper/ubuntu--vg-root ro
```

Listing 3.51 Standard-Kernelparameter aller verwendeten Distributionen

3.4.5 Gemountete Dateisysteme

Der Kernel muss natürlich auch wissen, welche Dateisysteme aktuell eingebunden sind. Diese Information wird in der virtuellen Datei */proc/mounts* zur Verfügung gestellt. Sie ist aber auch als Ausgabe des `mount`-Kommandos oder in der Datei */etc/mstab* zu finden:

```
opensuse:~ # grep mapper /proc/mounts
/dev/mapper/system-root / ext4 rw,relatime,data=ordered 0 0
```

```
opensuse:~ # mount -v | grep mapper
/dev/mapper/system-root on / type ext4 (rw,relatime,data=ordered)
```

```
opensuse:~ # grep mapper /etc/mtab
/dev/mapper/system-root / ext4 rw,relatime,data=ordered 0 0
```

Listing 3.52 Gemountete Dateisysteme

3

3.4.6 Prozessinformationen

Im Verzeichnis `/proc` finden Sie für jeden auf Ihrem Linux-System laufenden Prozess ein Unterverzeichnis unter dem Namen der Prozess-ID. Diese Informationen werden von vielen Tools benutzt. Als Beispiele seien hier `ps` (»process stats«) und `lsdf` (»list open files«) genannt.

```
root@debian:~# ls /proc/26803
attr          coredump_filter  io              mountstats     pagemap        stat
autogroup    cpuset           limits         net            personality    statm
auxv         cwd              loginuid       ns             root           status
cgroup       environ         maps           numa_maps     sched          syscall
clear_refs  exe              mem            oom_adj        sessionid      task
cmdline     fd               mountinfo      oom_score     smaps          wchan
comm        fdinfo           mounts         oom_score_adj stack
```

Listing 3.53 Prozessinformationen eines Prozesses

File Descriptors finden Sie für jeden Prozess im Unterverzeichnis `fd`. Dort sind symbolische Links für jede benutzte Datei zu finden. In Listing 3.54 ist es der PDF-Betrachter *okular*, der eine Datei geöffnet hat:

```
root@debian:~# ls -l /proc/26803/fd
total 0
lr-x----- 1 dirk dirk 64 Aug 17 17:42 0 -> pipe:[10146]
l-wx----- 1 dirk dirk 64 Aug 17 17:42 1 -> /home/dirk/.xsession-errors
lr-x----- 1 dirk dirk 64 Aug 17 17:42 10 -> anon_inode:inotify
lr-x----- 1 dirk dirk 64 Aug 17 17:42 11 -> /home/dirk/Downloads/Anleitung.pdf
l-wx----- 1 dirk dirk 64 Aug 17 17:42 2 -> /home/dirk/.xsession-errors
lrwx----- 1 dirk dirk 64 Aug 17 17:42 3 -> anon_inode:[eventfd]
lr-x----- 1 dirk dirk 64 Aug 17 17:42 4 -> pipe:[185471]
l-wx----- 1 dirk dirk 64 Aug 17 17:42 5 -> pipe:[185471]
lrwx----- 1 dirk dirk 64 Aug 17 17:42 6 -> socket:[185472]
lrwx----- 1 dirk dirk 64 Aug 17 17:42 7 -> socket:[185194]
lrwx----- 1 dirk dirk 64 Aug 17 17:42 8 -> socket:[185195]
lr-x----- 1 dirk dirk 64 Aug 17 17:42 9 -> /var/tmp/kdecache-dirk/ksycoca4
```

Listing 3.54 Geöffnete Dateien des PDF-Betrachters

Weiterhin finden Sie unterhalb des Unterverzeichnisses mit der Prozess-ID die virtuelle Datei `environ`, die das Environment enthält, mit dem der Prozess gestartet wurde.

Beachten Sie bitte, dass die einzelnen Shell-Variablen durch den »Null Character« getrennt sind und dass Sie die Ausgabe vor der Sichtung umformatieren müssen. Das kann das Programm `tr` (»translate or delete characters«) erledigen, wie Listing 3.55 zeigt:

```
root@debian:~# tr "\0" "\n" < /proc/26803/environ
KDE_FULL_SESSION=true
DM_CONTROL=/var/run/xdmctl
GS_LIB=/home/dirk/.fonts
USER=dirk
LANGUAGE=en_US:en
[...]
```

Listing 3.55 Lesbarer Inhalt von »environ«

Weiterhin findet sich in der virtuellen Datei `cmdline` der Aufruf des Programms mit allen verwendeten Parametern. Auch hier wird der »Null Character« zur Trennung benutzt:

```
root@debian:~# tr "\0" "\n" < /proc/26803/cmdline
/usr/bin/okular
--icon
okular
-caption
Okular
```

Listing 3.56 Lesbarer Inhalt von »cmdline«

3.4.7 Netzwerk

Die das Netzwerk betreffenden Informationen – Netzwerkkarten, Protokolle, Module etc. – sind sehr vielfältig und liegen in einigen Hundert Dateien unterhalb von `/proc/net` und `/proc/sys/net`. Wir können im Rahmen dieses Buches leider nicht alle Dateien beschreiben.

Fast alle Parameter rund um TCP (*Transmission Control Protocol*) lassen sich dynamisch verändern. So können Sie beispielsweise durch das in Listing 3.57 gezeigte Beispiel das Routing auf Ihrem System einschalten:

```
root@debian:~# echo "1" > /proc/sys/net/ipv4/ip_forward
```

Listing 3.57 Routing einschalten

Unterhalb von `/proc/sys/net/ipv4/conf` finden Sie zu jeder Netzwerkkarte die *IPv4* betreffenden Konfigurationen:

```
root@debian:~# ls /proc/sys/net/ipv4/conf
all default eth0 lo sit0 wlan0
```

Listing 3.58 Netzwerkkarten unterhalb von »ipv4«

Einen Sonderfall nehmen dabei die Verzeichnisse `all` und `default` ein. Mit ihnen können Sie Optionen setzen, die für alle Netzwerkkarten gelten sollen (`all`), bzw. Optionen, die für neue Netzwerkkarten gelten sollen (`default`).

3.4.8 Änderungen dauerhaft speichern

Wie eingangs erwähnt wurde, ist `/proc` ein virtuelles Dateisystem. Das bedeutet insbesondere, dass Änderungen, die Sie dort vornehmen, einen Neustart des Systems nicht überstehen.

Dauerhafte Konfigurationen sollten in Konfigurationsdateien im Verzeichnis `/etc/sysctl.d` vorgenommen werden. Hier können alle Kerneleinträge aufgenommen werden, die unterhalb von `/proc/sys` angezeigt werden.

Wie Sie in Listing 3.59 sehen, ist die Syntax nahezu selbsterklärend. Eine Besonderheit gibt es jedoch: Wenn Sie den Pfad zum Parameter gefunden haben, entfernen Sie das `/proc/sys/` und ersetzen einfach alle Slashes durch Punkte, dann haben Sie den Variablennamen für die `/etc/sysctl.conf`. Aus `/proc/sys/net/ipv4/ip_forward` wird somit `net.ipv4.ip_forward`:

```
root@debian:~# grep ip_forward /etc/sysctl.d/forward.conf
net.ipv4.ip_forward=1
```

Listing 3.59 Das Forwarding in der Datei »forward.conf« dauerhaft eintragen

Mit dem Kommando `sysctl -p` können Sie alle Einstellungen, die Sie vorgenommen haben, sofort und ohne Neustart übernehmen.

Das Kommando `sysctl -a` zeigt Ihnen die aktuellen Werte aller knapp 1.000 änderbaren Parameter an.

3.4.9 Abschlussbemerkung

Indem Sie die Dateien unterhalb von `/proc` studieren, können Sie sehr viel über Ihr laufendes System erfahren. Im Normalfall werden Sie jedoch vermutlich Tools verwenden, die Ihnen die Daten aufbereitet darstellen. Mit jeder neuen Kernelversion kommen neue Parameter hinzu und alte Parameter fallen weg. Einen umfassenden Überblick können Sie nur bekommen, wenn Sie die Kerneleldokumentation direkt lesen.

Kapitel 4

Dateisysteme

In diesem Kapitel lernen Sie die Arbeitsweise der wichtigsten Linux-Dateisysteme sowie den Umgang mit den dazugehörigen Tools kennen.

Was genau macht eigentlich ein Dateisystem, und warum gibt es so viele? Als Linux-Admin haben Sie die Wahl zwischen den Dateisystemen Ext2, Ext3 und Ext4, ReiserFS, XFS, JFS und BtrFS – und das sind nur die gängigsten, universell einsetzbaren Dateisysteme. Dieses Kapitel informiert Sie im Grundlagenteil über die Struktur der Dateisysteme und über die Unterschiede zwischen ihnen. Der Praxisteil erläutert den Umgang mit den dazugehörigen Tools in der täglichen Administrationsarbeit.

4.1 Dateisysteme: von Bäumen, Journalen und einer Kuh

Dateisysteme speichern Daten. Aber sie speichern nicht nur Daten, sondern auch Daten über Daten. Dazu gehören die Größe, der Besitzer oder Informationen darüber, welche Benutzer Zugriffsrechte auf die Daten besitzen. Diese »Daten über Daten« heißen *Metadaten*.

Das Dateisystem speichert die Daten und Metadaten auch nicht nur, es *verwaltet* sie. Essenzielle Verwaltungsaufgaben, die jedes Dateisystem beherrschen muss, sind das

- ▶ Finden,
- ▶ Einfügen und
- ▶ Löschen.

Komplexere Verwaltungsvorgänge wie das *Ändern* setzen sich aus Abfolgen von *Einfügen* und *Löschen* zusammen. Wie schnell, sicher und effizient ein Dateisystem ist, hängt maßgeblich davon ab, wie gut diese grundlegenden Funktionen implementiert sind.

Frühere, heute kaum noch gebräuchliche Dateisysteme verwalteten die Informationen darüber, wo eine bestimmte Datei auf dem Datenträger zu finden war, in einer einfachen Liste. Die Elemente der Liste enthielten Zeiger auf den gesuchten Datenblock. Diese Liste (*File Allocation Table*, FAT) zu verwalten war zwar einfach, aber nicht effizient. Um ein bestimmtes Element zu finden, musste im Extremfall die ganze Liste linear durchsucht werden. Bei einer relativ geringen Anzahl von Dateien sind die Nachteile zu verschmerzen, aber der rasante Anstieg der Festplattenkapazitäten machte bessere Ordnungssysteme erforderlich. Auf sehr

kleinen und entfernbaren Datenträgern wie CDs und einigen USB-Sticks haben die Dateisysteme des alten Typs bis heute überlebt, da ihre geringe Effizienz hier kaum ins Gewicht fällt.

4.1.1 Bäume

Die Organisationsstruktur, mit der moderne Dateisysteme die gespeicherten Daten wiederfinden, ähnelt einem Baum (siehe Abbildung 4.1). Der Baum hat eine Wurzel, Gabelungen und Blätter. Die Gabelungen werden *Knoten* genannt.

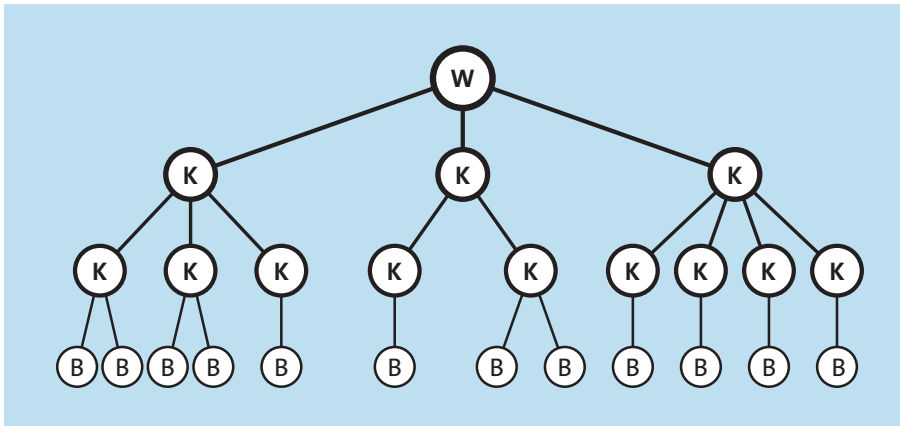


Abbildung 4.1 Schematische Darstellung einer Baumstruktur mit (W)urzel, (K)noten und (B)lättern

Von der Wurzel aus kann man über verhältnismäßig wenige Knoten jedes Blatt erreichen. Die maximale Anzahl der Schritte vom ersten Knoten (die Wurzel wird nicht mitgezählt) bis zu einem Blatt ist die *Höhe* des Baums. Ein Baum mit drei Knotenebenen hat also die Höhe 4 – drei Knotenebenen plus die Blattebene.

Baumarten und Suchstrategien

Um von der Wurzel aus möglichst schnell an jeden beliebigen Punkt des Baums zu gelangen, muss der Baum nicht hoch, sondern möglichst breit sein.¹ Daraus folgt automatisch, dass der Baum möglichst »in der Balance« sein muss, das heißt: Es soll keinen Zweig mit sieben Knoten geben, während ein anderer Zweig nur zwei Knoten hat. Außerdem soll die Höhe des Baums sich in allen Zweigen widerspiegeln. Abbildung 4.2 zeigt als Negativbeispiel einen unbalancierten Baum.

¹ In einem Dateisystem kann ein Knoten durchaus 200 Unterknoten haben.

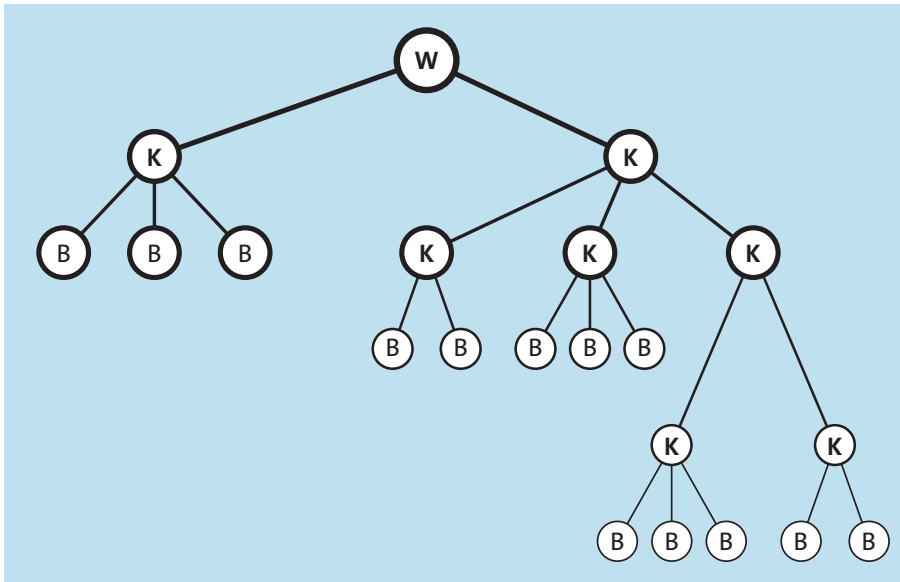


Abbildung 4.2 Eine unbalancierte Baumstruktur

Wurzel, Knoten und Blätter stehen hier natürlich nur metaphorisch für Datenstrukturen. Sie enthalten Schlüssel, anhand derer man sie innerhalb des Baums identifizieren kann. Die Datenstrukturen können neben den Schlüsseln Daten, Metadaten oder beides enthalten. Durch diese Varianz entstehen mehrere Baumarten:

► **B-Bäume**

B-Bäume sind per Definition balanciert: Alle Blätter liegen auf der gleichen Ebene. Ist die Balance durch Dateioperationen wie Löschen oder Einfügen gefährdet, wird sie durch Neuorganisation von Blättern wiederhergestellt. Dieses Verfahren heißt *Rotation*. Ein weiteres Merkmal von B-Bäumen ist, dass Daten sowohl in den Knoten als auch in den Blättern gespeichert werden. Ext3 und Btrfs benutzen B-Bäume; Abbildung 4.1 ist eine schematische Darstellung eines B-Baums.

► **B+-Bäume**

In einem B+-Baum werden Daten ausschließlich in den Blättern gespeichert, nicht in den Knoten. Davon profitieren insbesondere Löschoperationen, da Rotationen über Ebenengrenzen hinweg nicht auftreten können. Dateisysteme, die B+-Bäume benutzen, sind beispielsweise ReiserFS 3.x und XFS.

► **H-Bäume**

Bei einem H-Baum steht die maximale Anzahl der Knoten und Blätter mit dem Anlegen des Dateisystems fest. Das bedeutet einen Verzicht auf Flexibilität, eliminiert aber auch die Notwendigkeit aufwendiger Reorganisationen. Abbildung 4.3 zeigt den schematischen Aufbau eines H-Baums.

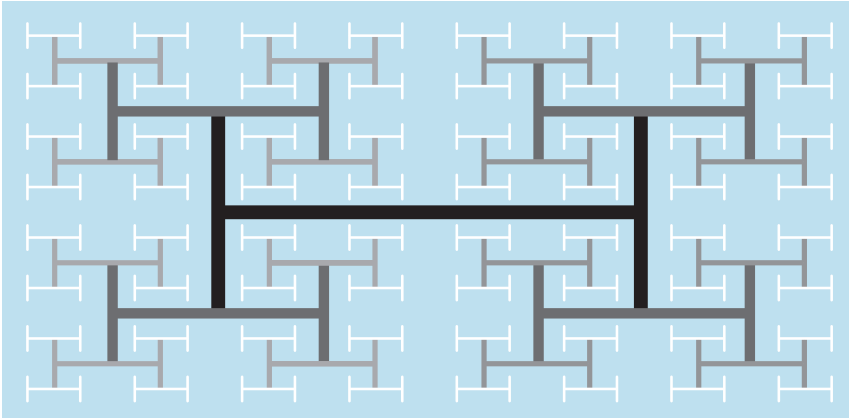


Abbildung 4.3 Schematische Darstellung eines H-Baums

4.1.2 Journale

Wenn Sie eine Datei anlegen, löschen, verschieben oder ändern, sind stets mehrere Datenbereiche auf einer Festplatte involviert: Schlüssel, Metadaten und die Daten selbst, die sich bei großen Dateien auch über mehrere, nicht zusammenhängende Blöcke erstrecken können.

Das Dateisystem hat also ein Problem. Nachdem es die erste Schreiboperation ausgeführt hat, ist der Datenbestand nicht mehr konsistent. Er ist erst dann wieder konsistent, wenn die letzte Schreiboperation erfolgreich abgeschlossen wurde. Fällt innerhalb dieser Zeitspanne der Strom aus, muss das Dateisystem beim Neustart durch eine aufwendige Prüfung und Reparatur wieder einen konsistenten Zustand herstellen. Dieser Vorgang ist auf großen Dateisystemen sehr zeitintensiv. Abhilfe schafft ein *Journaling File System*. Bevor das System die Schreibzugriffe ausführt, schreibt es in das Journal, was es zu tun gedenkt. In bestimmten Zeitabständen wird das Journal abgeschlossen.

Das bedeutet, die »To-do-Liste« im Journal wird mit den tatsächlich durchgeführten Aktionen abgeglichen und aktualisiert. Kommt es zu einem Ausfall, kann zwar immer noch Datenverlust auftreten – nämlich bei den Daten, die zwischen zwei Journal-Abschlusszyklen verändert wurden –, aber Inkonsistenzen werden vermieden, und das System kann ohne langwierige Prüfungen wieder starten.

4.1.3 Und die Kühe? COW-fähige Dateisysteme

Beim Speichern von Daten wird der ursprüngliche Plattenbereich mit den neuen Daten überschrieben. Reicht der Platz nicht mehr aus, werden die zusätzlichen Daten in einen freien Speicherbereich geschrieben. Das dazu notwendige Neupositionieren des Schreib-/Lesekopfs der Platte kostet wertvolle Zeit. Besser ist es, die Daten gleich zusam-

menhängend in einen freien Speicherbereich zu schreiben. So gewinnt man nicht nur Zeit, sondern erhält gratis noch eine Versionsverwaltung, denn die alten Daten sind ja noch auf der Platte. Diese Technik nennt sich »COW« und hat nichts mit wiederkäuenden Paarhufern zu tun, sondern steht für *Copy on Write*. COW-fähige Dateisysteme sind beispielsweise ZFS und BtrFS.

4.2 Praxis

Nach der Lektüre des Grundlagenteils haben Sie sicher schon ein Gespür dafür bekommen, wie die interne Organisationsstruktur eines Dateisystems sein Verhalten in der Praxis beeinflusst. Im Praxisteil lernen Sie den Umgang mit den Verwaltungswerkzeugen der Dateisysteme Ext2/3, ReiserFS und XFS.

4.2.1 Ext2/3-FS aufgebohrt: mke2fs, tune2fs, dumpe2fs, e2label

Die Dateisysteme der Ext-Familie werden von den meisten Linux-Distributionen als Standarddateisystem eingesetzt, weil sie als besonders robust gelten. Bereits beim Anlegen des Ext-Dateisystems können Sie es für seine zukünftigen Aufgaben optimieren.

Ein Ext-Dateisystem anlegen

Wenn Sie ein universell einsetzbares Filesystem benötigen, das gleichermaßen gut mit großen wie kleinen Dateien zurechtkommen soll, legen Sie es einfach mit dem folgenden Kommando an:

```
mke2fs /dev/<Gerätename>
```

Listing 4.1 Ext-Dateisystem ohne weitere Optionen anlegen

`mke2fs` wird abhängig von der Größe der Partition sinnvolle Default-Werte für die Anzahl der Inodes und für die Blockgröße wählen. Es geht dabei davon aus, dass das Dateisystem eine Mischung aus großen, mittleren und kleinen Dateien enthalten wird. Falls Sie allerdings bereits wissen, dass Ihr neues Dateisystem künftig mehrheitlich sehr kleine (Mail-Server) oder sehr große (Videoschnitt, ISO-Images) Dateien aufnehmen können soll, können Sie mit dem Parameter `-T` auf das Layout des Dateisystems Einfluss nehmen (siehe Tabelle 4.1).

Wert	Blockgröße	Inodes pro x Blöcke	Nutzung
small	1 KByte	x = 4	für kleine Partitionen (Default bis 512 MB)
news	4 KByte	x = 1	für viele kleine Dateien

Tabelle 4.1 Optimierung des Dateisystems auf bestimmte Aufgaben

Wert	Blockgröße	Inodes pro x Blöcke	Nutzung
largefile	4 KByte	x = 256	für große Dateien, 1 MB pro Inode
largefile4	4 KByte	x = 1.024	für große Dateien, 4 MB pro Inode

Tabelle 4.1 Optimierung des Dateisystems auf bestimmte Aufgaben (Forts.)

Die Routineüberprüfung

Nach einer bestimmten Anzahl von Mountvorgängen oder nach Ablauf einer definierten Zeitspanne – je nachdem, was zuerst eintritt – werden Ext-Dateisysteme einer Routineüberprüfung unterzogen. Auf Systemen, die oft neu gestartet werden, können Sie diese Werte mit dem Kommando `tune2fs` erhöhen. So erfolgt die Prüfung nach 50 Mounts oder 60 Tagen:

```
tune2fs -c 50 -i 60
```

Listing 4.2 Prüfintervall heraufsetzen

Auf- und Abwärtskompatibilität

Das Dateisystem Ext3 (*Third Extended Filesystem*) erweitert seinen Vorgänger Ext2 um die Journaling-Fähigkeit, die Inkonsistenzen zu vermeiden hilft. Beide haben eine hohe Stabilität und Robustheit, die über die Jahre durch schrittweise Verbesserungen und den massenhaften Einsatz auf unzähligen Linux-Systemen erreicht wurde. Beide Systeme sind miteinander kompatibel. Ein Ext3-Dateisystem kann als Ext2 gemountet werden und arbeitet mit Ausnahme des abgeschalteten Journals so, wie Sie es von einem Ext2-System gewohnt sind. Umgekehrt können Sie ein Ext2-Dateisystem jederzeit mit einem Ritterschlag zum Ext3-System aufwerten, so wie in Listing 4.3 dargestellt:

```
tune2fs -j /dev/sda3
```

Listing 4.3 Ext2-Filesystem in Ext3 konvertieren

Prinzipiell funktioniert das sogar, während das Dateisystem gemountet ist – das Journal würde in diesem Fall in einer sichtbaren Datei mit dem Namen `.journal` angelegt. Aber sicherer ist es, die Konvertierung am ausgehängten Dateisystem vorzunehmen.

Journaling

Ext3 kennt zwei Journaling-Methoden, um die Konsistenz der Metadaten zu schützen, und eine dritte, die auch die eigentlichen Nutzdaten mit einbezieht:

► `data=ordered`

Dies ist die Standardeinstellung, ein Kompromiss zwischen Sicherheit und Geschwindigkeit. Erst nachdem die Nutzdaten einer Schreiboperation auf die Platte geschrieben wurden, werden die Journaleinträge aktualisiert.

► **data=writeback**

Das Dateisystem wartet nicht mit dem Aktualisieren des Journals, bis die Nutzdaten tatsächlich geschrieben wurden. Diese Methode bringt einen Geschwindigkeitsvorteil, kann aber zu Datenverlust führen, wenn zum Zeitpunkt eines Ausfalls das Journal bereits aktualisiert war, aber die Daten noch nicht vollständig geschrieben wurden.

► **data=journal**

Im Gegensatz zum *Metadata Journaling* der Methoden *ordered* und *writeback* etabliert diese Methode das *Full Journaling*. Daten und Metadaten werden zunächst in das Journal geschrieben. Erst nachdem die Daten von dort aus an ihre endgültige Position im Dateisystem kopiert wurden, wird das Journal aktualisiert. Weil auf diese Weise alle Schreibvorgänge doppelt ausgeführt werden, ist das *Full Journaling* bei Schreiboperationen recht langsam, vermeidet aber wirkungsvoll Dateninkonsistenzen.

Sie können insbesondere die Schreibperformance verbessern, indem Sie das Journal auf einen separaten Datenträger schreiben lassen, etwa einen kleinen RAID-Verbund. Dazu gehen Sie in zwei Schritten vor: Zuerst bereiten Sie das Block-Device vor, auf dem das Journal liegen soll, und im zweiten Schritt formatieren Sie die eigentliche Datenplatte und übergeben als Parameter das Journalgerät. Wenn Ihre Datenplatte `/dev/sda1` und Ihre Journalplatte `/dev/sdb1` heißt, lauten die beiden Kommandos wie folgt:

```
# Journaldatenträger vorbereiten:
mke2fs -O journal_dev /dev/sdb1

# Datenplatte formatieren, auf Journal-Device verweisen:
mke2fs /dev/sda1 -J device=/dev/sdb1
```

Listing 4.4 Dateisystem mit separatem Journal-Device

Zusätzlich können Sie mit dem Parameter `-L <Bezeichnung>` jedem Dateisystem einen Bezeichner (*Label*) zuweisen. Das ist auch nachträglich mit dem Befehl `e2label <Gerät> <Bezeichner>` möglich. Das *Label* darf nicht mehr als 16 Zeichen lang sein.

dumpe2fs: Informationen über das Filesystem

Mit `dumpe2fs /dev/<Gerätename>` lassen Sie sich Informationen über das Ext-Dateisystem ausgeben. Die Ausgabe ist recht lang, da auch alle Informationen über Superblocks und Blockgruppen ausgegeben werden. In den meisten Fällen reicht es aus, wenn Sie sich die verkürzte Ausgabe mit dem Kommando `dumpe2fs -h /dev/<Gerätename>` ausgeben lassen.

Hier sehen Sie unter anderem den aktuellen Status des Dateisystems und Informationen über den zyklischen *File System Check*:

```
[...]
Filesystem state:          clean
```

```
[...]  
Filesystem created:      Wed Nov 11 10:25:43 2009  
Last mount time:       Sun Sep 26 12:51:25 2010  
Last write time:      Tue May 25 14:24:51 2010  
Mount count:          14  
Maximum mount count:   31  
Last checked:         Tue May 25 14:24:51 2010  
Check interval:       15552000 (6 months)  
Next check after:     Sun Nov 21 13:24:51 2010  
Lifetime writes:      3086 MB  
[...]
```

Listing 4.5 Ausgabe von »dumpe2fs« (Auszug)

Sollten auf Ihrem Dateisystem einige Blöcke als *bad* (schlecht, beschädigt) markiert sein, können Sie sich mit dem Befehl `dumpe2fs -b` ausgeben lassen, welche Blöcke betroffen sind.

Dateisystemprüfung mit »e2fsck«

Wenn Sie den Verdacht haben, dass Ihr Dateisystem über noch nicht lokalisierte *bad blocks* stolpert, rufen Sie einmal `e2fsck -c /dev/<Gerätename>` auf. Das Dateisystem darf dabei nicht gemountet sein. `e2fsck` durchsucht das Dateisystem mithilfe des externen Programms `badblocks` nach Schädstellen. Die IDs der kaputten Blöcke werden einem speziellen Inode hinzugefügt und vom Dateisystem künftig gemieden.

Fehler, die nicht auf schlechten Blöcken beruhen, können Sie ebenfalls bei ausgehängtem Dateisystem mit dem Befehl `e2fsck -p /dev/<Gerätename>` orten. Das Programm wird versuchen, alle gefundenen Fehler ohne weitere Interaktion Ihrerseits zu beheben. Ist das nicht möglich, wird eine Beschreibung des Fehlers ausgegeben, und die Verarbeitung stoppt.

4.2.2 ReiserFS und seine Tools

ReiserFS war 2001 das erste Journaling-Dateisystem, das in den offiziellen Linux-Kernel einzog, damals allerdings noch in experimentellem Zustand. »Produktionsreif« wurde ReiserFS erst später. ReiserFS kann kleine Dateien etwas effizienter speichern als andere Dateisysteme und bietet dadurch auf einem ansonsten gleichen *Volume* etwa 5% mehr Speicherkapazität. Die höhere Packungsdichte bedingt aber eine etwas geringere Schreibgeschwindigkeit und lässt sich deshalb abschalten, indem in der `/etc/fstab` der Parameter `notail` gesetzt wird.

ReiserFS skaliert auf Systemen mit Mehrkernprozessoren nicht optimal, weil nicht alle Teile des Codes auf mehreren Kernen parallel ausgeführt werden können. Entwicklungen, die dieses Problem beheben, sind zwar in Arbeit, aber noch nicht in den offiziellen Linux-Kernel eingeflossen.

Ein ReiserFS-Dateisystem anlegen

Der folgende Befehl formatiert ein Block-Device mit dem Reiser-Dateisystem:

```
mkfs.reiserfs /dev/<Gerätename>
```

Listing 4.6 Ein ReiserFS-Dateisystem ohne weitere Optionen anlegen

Sie können den Befehl um `-l <Bezeichner>` ergänzen, um dem Dateisystem ein *Label* anzuhängen. Wie bei den Ext-Dateisystemen darf das Label nicht mehr als 16 Zeichen umfassen. Auch ReiserFS erlaubt es, das Journal auf einen separaten Datenträger zu legen. Wenn Ihre Daten auf dem Gerät `/dev/sda1` und das Journal auf `/dev/sdb1` liegen sollen, lautet das Kommando:

```
mkfs.reiserfs /dev/sda1 -j /dev/sdb1
```

Listing 4.7 ReiserFS-Dateisystem mit ausgelagertem Journal anlegen

Sie können das Journal auch nachträglich auf einen anderen Datenträger auslagern. Dazu benötigen Sie den Befehl `reiserfstune`:

```
reiserfstune /dev/sda1 --journal-new-device /dev/sdb1
```

Listing 4.8 Journal nachträglich auf ein anderes Device legen

4.2.3 XFS

XFS wurde in den 90er-Jahren von *Silicon Graphics* entwickelt und steht seit 2001 für Linux zur Verfügung. Schreiboperationen auf das Journal laufen unabhängig von den eigentlichen Datentransaktionen ab, die XFS möglichst lange im RAM puffert (*Delayed Allocation*). Auf diese Weise wird der Durchsatz erhöht und Fragmentierung beim Schreiben vermieden. Dadurch wird zwar auch die Gefahr eines Datenverlusts bei einem Totalausfall des Systems erhöht, die Konsistenz des Dateisystems ist jedoch sichergestellt.

Ein XFS-Dateisystem anlegen

Wie Ext und ReiserFS kann auch XFS sein Journal auf das gleiche Blockgerät schreiben wie die Nutzdaten oder ein »externes« Journal führen. So legen Sie das Dateisystem an:

```
# XFS mit internem Journal (Journal und Daten auf gleicher Partition)
mkfs.xfs /dev/sda1
```

```
# XFS mit externem Journal (Journal auf separater Partition)
mkfs.xfs -l logdev=/dev/sdb1 /dev/sda1
```

Listing 4.9 XFS-Filesystem mit internem und externem Journal

Das Dateisystem reorganisieren

Sollte das Dateisystem einmal merklich langsamer werden, können Sie es mit dem Befehl `xfs_fsr` reorganisieren. Geben Sie den Befehl einfach als `root` auf einer Kommandozeile ein – er funktioniert auch online, also während das Dateisystem gemountet ist. `xfs_fsr` ermittelt, welche Dateien besonders stark fragmentiert sind, und beginnt dort mit der Neuorganisation. Per Default bricht das Programm nach zwei Stunden automatisch ab, kann aber manuell neu gestartet werden und fährt an der Stelle fort, an der es aufgehört hat. Sie können auch mit dem Parameter `-t <Laufzeit_in_Sekunden>` eine andere Arbeitsdauer festlegen. Es empfiehlt sich, die Neuorganisation *cron*-gesteuert zu einer Zeit vornehmen zu lassen, in der das Dateisystem wenig belastet ist.

4.2.4 Das Dateisystem vergrößern oder verkleinern

Sie können das Dateisystem bei Bedarf vergrößern oder – im Fall von Ext2/3 und ReiserFS – auch verkleinern. Ein XFS-Dateisystem lässt sich hingegen nicht verkleinern. Beim Vergrößern eines Dateisystems muss die Partition natürlich eine gewisse Reserve aufweisen, die das Dateisystem nutzen kann. Liegt das Dateisystem auf einer logischen Partition, erweitern Sie diese zunächst mit dem `lvextend`-Kommando (siehe Abschnitt 3.2, »Rein logisch: Logical Volume Manager ›LVM«).

Eine Änderung der Größe ist immer ein recht tiefer Eingriff ins Dateisystem. Obwohl die Tools einen hohen Reifegrad haben, sollten Sie über ein aktuelles Backup verfügen. Vorher noch einen *File System Check* laufen zu lassen, ist ebenfalls eine gute Idee.

Die Größe eines Ext-Dateisystems ändern

Ext-Dateisysteme können Sie online und offline, also im eingehängten oder im nicht eingehängten Zustand, vergrößern, aber nur offline verkleinern. Wenn Sie Ihr Dateisystem auf die maximale Größe der (logischen) Partition ausdehnen möchten, geben Sie als `root` diesen Befehl ein:

```
resize2fs -p /dev/<Gerätename>
```

Listing 4.10 Ext-Dateisystem vergrößern

Der Parameter `-p` erzeugt einen Fortschrittsbalken, während die Vergrößerung läuft. Eine Verkleinerung nehmen Sie vor, indem Sie die Zielgröße des Dateisystems angeben. Denken Sie daran, dass Sie Ext-Dateisysteme nur im ausgehängten Zustand verkleinern können. Falls Ihr System diese Partition zum Starten benötigt, verwenden Sie eine Live-CD wie *Knoppix*.

Wenn Ihr Dateisystem 80 GB groß ist und auf 60 GB schrumpfen soll, lautet das Kommando:

```
resize2fs -p /dev/<Gerätename> 60G
```

Listing 4.11 Ext-Dateisystem verkleinern

Die Größe eines Reiser-Dateisystems ändern

Ein Reiser-Dateisystem lässt sich, sofern das Dateisystem nicht gemountet ist, vergrößern und verkleinern. Wie bei den Ext-Dateisystemen können Sie auch eine bestimmte Zielgröße angeben, auf die das Dateisystem vergrößert oder verkleinert werden soll:

```
resize_reiserfs /dev/<Gerätename> 60G
```

Listing 4.12 ReiserFS auf eine bestimmte Zielgröße bringen

Ebenso können Sie das vorhandene Dateisystem auf die maximale Größe aufblähen, die die (logische) Partition hergibt. Dazu lassen Sie einfach die Angabe der Zielgröße weg:

```
resize_reiserfs /dev/<Gerätename>
```

Listing 4.13 ReiserFS maximal vergrößern

Alternativ haben Sie die Möglichkeit, die Differenz zur aktuellen Größe des Filesystems anzugeben:

```
# Dateisystem um 20 GB verkleinern:
resize_reiserfs /dev/<Gerätename> -20G
```

```
# Dateisystem um 20 GB vergrößern:
resize_reiserfs /dev/<Gerätename> +20G
```

Listing 4.14 ReiserFS relativ verändern

Die Größe eines XFS-Systems ändern

Ein XFS-System können Sie lediglich vergrößern, nicht verkleinern. Die Änderung der Größe funktioniert nur, während das Dateisystem gemountet ist.

Eine Vergrößerung auf eine von Ihnen definierte Größe ist nicht möglich, XFS vergrößert sich immer auf die gesamte zur Verfügung stehende Partitionsgröße.

Im Unterschied zu den anderen Dateisystemen geben Sie bei XFS nicht das (logische) Gerät, sondern den Mountpunkt an:

```
xfs_growfs /<Mountpunkt> -o remount,resize
```

Listing 4.15 XFS vergrößern

4.2.5 BtrFS

Der Name *BtrFS* deutet auf die verwendeten B-Bäume hin (*B-Tree-FS*), wird aber meist wie »Butter-FS« oder »Better-FS« ausgesprochen. Obwohl die Entwicklung noch nicht vollständig abgeschlossen ist, insbesondere was die Filesystem-Tools angeht, erhält BtrFS bereits viel

Lob. Die Wahrscheinlichkeit ist hoch, dass BtrFS in nicht ferner Zukunft die Nachfolge der Ext-Dateisysteme als Quasi-Standard antreten wird.



Sehr lange fand sich im offiziellen Getting-Started-Guide des BtrFS-Wiki² der Warnhinweis, dass man den aktuellsten Kernel verwenden sollte, wenn man BtrFS einsetzt. Weiterhin war dort der Hinweis zu finden, dass man ein Backup besitzen und darauf vorbereitet sein sollte, es einzusetzen. Seit 2014 gilt das Dateisystem als stabil, regelmäßige Backups sollten Sie natürlich trotzdem erstellen.

In der Tat ist die Liste der unterstützten Funktionen beeindruckend. Als COW-System unterstützt BtrFS Snapshots. Der Zustand des Dateisystems wird dabei in einem Subvolume eingefroren und kann dort anderweitig genutzt werden, etwa für ein Backup. Solid-State-Speicher werden gesondert unterstützt, außerdem ist die Unterstützung der RAID-Level 0, 1 und 10 implementiert. Die Level 5 und 6 gelten immer noch als experimentell, und das schon seit Jahren. Wie XFS kann BtrFS im gemounteten Zustand defragmentiert werden. Außerdem erkennt BtrFS, wenn es auf einer SSD eingesetzt wird, und aktiviert dafür selbstständig sinnvolle Einstellungen.

Die große Stärke von BtrFS ist, dass es komplette Devices verwalten und durch Subvolumes gekapselte Mountpunkte exportieren kann. Im kommerziellen Bereich bietet das sehr ausgereifte, ZFS ähnliche Möglichkeiten. Da die Lizenz von ZFS (Common Development and Distribution License, CDDL) nicht kompatibel mit der Lizenz des Linux-Kernels (GNU Public License, GPL) ist, wird sich ZFS nie im Linux-Kernel befinden. Aus diesem Grund versuchen namhafte Firmen die Entwicklung von BtrFS nach vorn zu treiben, um die Features eines modernen Dateisystems mit integriertem Volumemanager auch unter Linux direkt nutzen zu können.



Beachten Sie bitte, dass unter Debian und Ubuntu das Paket *btrfs-progs* und unter openSUSE das Paket *btrfsprogs* installiert sein sollte, wenn Sie mit BtrFS arbeiten wollen. In CentOS Stream gibt es derzeit keinen BtrFS-Support, was umso erstaunlicher ist, als dass Fedora mit Version 33 dieses Dateisystem als Standard für Desktops einsetzt.

Ein BtrFS-Dateisystem anlegen und die Komprimierung aktivieren

Da die Entwicklung nach wie vor andauert, sollten Sie BtrFS derzeit noch mit Vorsicht genießen. Das Kommando *btrfsfsck* existiert zwar, aber dahinter verbirgt sich derzeit nur ein Integritäts-Check, echte Reparaturen am Dateisystem können Sie damit noch nicht durchführen.

Wenn Sie BtrFS einmal ausprobieren möchten, können Sie es wie jedes andere Dateisystem auf einer freien Partition anlegen, im folgenden Beispiel verwenden wir */dev/sdb1*.

² https://archive.kernel.org/oldwiki/btrfs.wiki.kernel.org/index.php/Getting_started.html



Dieses Beispiel dient Demonstrationszwecken. Auch wenn es möglich ist, direkt auf das Filesystem innerhalb der Partition zuzugreifen, geht die Konzeption von BtrFS in die Richtung, dass Sie das nicht müssen, sondern stattdessen mit Subvolumes arbeiten, die später in diesem Abschnitt beschrieben werden.

```
mkfs.btrfs /dev/sdb1
```

Listing 4.16 Ein BtrFS-Dateisystem anlegen bzw. eine Partition BtrFS zur Verfügung stellen

Auch das Einhängen des neuen Dateisystems in den Verzeichnisbaum funktioniert wie bei anderen Dateisystemen. Wenn Sie möchten, können Sie BtrFS beim Mountbefehl zusätzlich anweisen, die Datenkomprimierung einzuschalten:

```
mount -o compress /dev/sdb1 /mnt/meinbtrfs
```

Listing 4.17 Das neue Dateisystem mounten und dabei die optionale Komprimierung einschalten

Die Komprimierung verschlingt natürlich einige CPU-Zyklen, aber in Zeiten schneller Multi-core-CPU's ist es unwahrscheinlich, dass Sie diese Verzögerung wahrnehmen. Im Gegenteil: Die Komprimierung wird sich eher positiv auf die Verarbeitungsgeschwindigkeit auswirken, weil Schreibzugriffe auf die Platten wesentlich flotter ablaufen.

Das gilt natürlich nicht für Daten, die bereits komprimiert sind, wie JPEG-Bilder, die meisten Videos und Musik im MP3-Format. BtrFS erkennt solche bereits komprimierten Dateien und macht gar keinen Versuch, sie noch weiter zu verdichten. Bei gut komprimierbaren Daten wie Text oder Quellcode ist der Gewinn an Geschwindigkeit und Plattenplatz allerdings beträchtlich.

Ein Ext2/3/4-Dateisystem in BtrFS konvertieren

Sie können ein mit Ext2, Ext3 oder Ext4 formatiertes Dateisystem nachträglich in BtrFS konvertieren. Dazu müssen Sie das Dateisystem (im folgenden Beispiel befindet es sich wieder auf der Partition */dev/sdb1*) zunächst aushängen:

```
# zuerst wird das Dateisystem ausgehängt
umount /dev/sdb1
```

```
# jetzt erfolgt die Konvertierung
btrfs-convert /dev/sdb1
```

```
# danach können Sie das Dateisystem wieder mounten (mit Komprimierung)
mount -o compress /dev/sdb1 /mnt/meinbtrfs
```

Listing 4.18 Eine Ext2/3/4-Partition in BtrFS konvertieren

Sie werden feststellen, dass die Konvertierung recht flott vonstattengeht, selbst auf großen Dateisystemen. Das liegt daran, dass der Konverter die Nutzdaten überhaupt nicht touchiert,

sondern lediglich die BtrFS-Metadaten schreibt und einen Snapshot des ursprünglichen Dateisystems erstellt. Wenn Sie entscheiden, dass die Konvertierung in BtrFS keine gute Idee war, können Sie den ursprünglichen Zustand jederzeit mithilfe des Snapshots wiederherstellen (siehe Listing 4.19). Dabei verlieren Sie allerdings alle Änderungen, die Sie seit der Konvertierung vorgenommen haben.

```
# wieder wird zuerst das Dateisystem ausgehängt
umount /dev/sdb1

# die Konvertierung rückgängig machen
btrfs-convert -r /dev/sdb1

# danach können Sie das Dateisystem wieder mounten
mount /dev/sdb1 /mnt/meinext3fs

# Kontrolle
df -h
```

Listing 4.19 Die Konvertierung rückgängig machen

Wenn Sie sich entschließen, BtrFS zu behalten, und auf die Möglichkeit der Rückkonvertierung keinen Wert legen, dann können Sie den Snapshot – er heißt `ext2_saved` – löschen:

```
btrfs subvolume delete /mnt/meinbtrfs/ext2_saved
```

Listing 4.20 Den Snapshot des alten Dateisystems entfernen

Subvolumes

Subvolumes werden oft als *Namespaces* innerhalb eines BtrFS-Dateisystems beschrieben. Es ist aber einfacher, sich Subvolumes als *virtuelle Dateisysteme* vorzustellen. Bei der Erstellung eines BtrFS-Dateisystems wird bereits ein *root-Subvolume* angelegt. Subvolumes können wieder weitere Subvolumes enthalten und so ineinander verschachtelt werden.

Wir haben bereits in der Partition `sdb1` ein Dateisystem erstellt, das wir nun nach `/mnt/btrfs` mounten. Dort erstellen wir ein Verzeichnis `home` und darin ein Subvolume pro User; `opt` soll ein weiteres Subvolume werden. Diese Subvolumes lassen sich selbstverständlich anzeigen, wie Sie in Listing 4.21 sehen. Leider wird das standardmäßig erstellte Subvolume für das Filesystem nicht angezeigt, aber da es immer die ID 5 hat, ist es leicht zu identifizieren.

```
# btrfs subvolume create /mnt/btrfs/home/user1
Create subvolume '/mnt/btrfs/home/user1'
# btrfs subvolume create /mnt/btrfs/home/user2
Create subvolume '/mnt/btrfs/home/user2'
# btrfs subvolume create /mnt/btrfs/opt
Create subvolume '/mnt/btrfs/opt'
```

```
# btrfs subvolume list -apt /mnt/btrfs
ID      gen      parent  top level  path
--      ---      -
257     6        5       5          home/user1
258     7        5       5          home/user2
259     8        5       5          opt
```

Listing 4.21 Erstellung von Subvolumes

Subvolumes verhalten sich innerhalb des Parent-Volumes wie Verzeichnisse, können allerdings nicht mit `rm` gelöscht werden. Dafür muss `btrfs subvolume delete` verwendet werden. Der Parameter ist der gleiche wie beim Erstellen der Subvolumes.

Eine Besonderheit von Subvolumes ist, dass sie auch eigenständig mittels `mount` eingebunden werden können. Dazu kann entweder der Name oder die ID des Subvolumes verwendet werden:

```
# mount -o subvolume=home/user1 /dev/sdb1 /tmp/1
# mount -o subvolid=257 /dev/sdb1 /tmp/1
```

Listing 4.22 Mounten von Subvolumes

Snapshots

Sie können jederzeit von beliebigen Subvolumes Snapshots erstellen. Diese Abbilder enthalten allerdings nur Daten des Subvolumes, für das sie erstellt wurden, nicht für eventuell darunter liegende Subvolumes. Anders als bei LVM-Snapshots (siehe Abschnitt 3.2.8 »Backups mit Snapshots«) müssen Sie bei BtrFS keine besonderen Vorbereitungen treffen. Es hat sich als hilfreich erwiesen, den Snapshots die Endung »-snap« zu geben, um sie leichter identifizieren zu können. Snapshots sind beschreibbar!

BtrFS-intern werden Snapshots wie Subvolumes behandelt. Der einzige Unterschied ist, dass sie zu Beginn keine Daten enthalten. Erst wenn sich das Ursprungs-Subvolume oder der Snapshot verändert, wird – gemäß dem Copy-on-Write-Prinzip – Speicherplatz verwendet. Der Parameter `-r` erzeugt beim Erstellen einen nur lesbaren Snapshot:

```
# btrfs subvolume snapshot /mnt/btrfs/opt /mnt/btrfs/opt-snap
# btrfs subvolume snapshot -r /mnt/btrfs/opt /mnt/btrfs/opt-snap-ro
# btrfs mount -o subvolume=opt-snap /dev/sdb1 /tmp/snap
# btrfs mount -o subvolume=opt-snap-ro /dev/sdb1 /tmp/snap-ro
```

Listing 4.23 Erstellung von Snapshots

Defragmentierung

Wie bei allen COW-Dateisystemen ist Fragmentierung bei BtrFS ein Problem. Geänderte Daten dürfen nicht einfach überschrieben werden, sondern wandern in freie Speicherberei-

che. Ist der Speicherbereich kleiner als die zu schreibenden Daten, werden diese zerstückelt, und es kommt zu Fragmentierung. Deshalb müssen BtrFS-Dateisysteme regelmäßig defragmentiert werden. Die Defragmentierung können Sie manuell vornehmen, indem Sie ein Verzeichnis angeben (tatsächlich ein Verzeichnis, nicht ein Subvolume), dessen Daten neu geordnet werden sollen:

```
sudo btrfs filesystem defragment /opt
```

Listing 4.24 Defragmentierung der Daten eines Verzeichnisses

Sie können während der Defragmentierung weiterarbeiten, aber das System wird sich etwas zäher anfühlen. Verzeichnisse auf SSDs müssen nicht defragmentiert werden – die Fragmentierung findet dort zwar auch statt, ist aber unproblematisch, da sie im Gegensatz zu Festplatten nicht zu einer Verringerung des Plattendurchsatzes führt: Eine SSD muss ja keine Schreib-/Leseköpfe neu positionieren. Sie können auch eine Hintergrund-Defragmentierung aktivieren. Dazu fügen Sie einfach in der */etc/fstab* die Mountoption `autodefragment` hinzu.

Mehrere Devices in BtrFS

Wie eingangs beschrieben wurde, sollte man beim Erstellen des BtrFS-Dateisystems davon sprechen, dass Devices BtrFS zur Verfügung gestellt werden. BtrFS bietet Unterstützung für verschiedene RAID-Level, die sich getrennt nach Nutzdaten und Metadaten, die zur Verwaltung von BtrFS verwendet werden, einstellen lassen. Die beiden gebräuchlichsten Fälle sind *Striping* (RAID-0) und *Mirroring* (RAID-1). Wenn man nur Devices ohne Angabe eines RAID-Levels zur Verfügung stellt, sorgt BtrFS für ein Striping der Nutzdaten und ein Mirroring der Metadaten. Um ein Filesystem mit RAID-Level zu mounten, müssen Sie nur ein Device des Verbunds angeben:

```
# mkfs.btrfs -f -d raid1 -m raid1 /dev/sdb /dev/sdc
# mount -o compress /dev/sdb /mnt/btrfs
```

Listing 4.25 Ein BtrFS mit RAID-1 anlegen

Bitte unterschätzen Sie das Volumen der Metadaten nicht! BtrFS kann einige seiner Vorteile nur durch die intensive Nutzung von Metadaten zur Verfügung stellen. Selbst wenn Ihnen das `df`-Kommando freien Speicher anzeigt, heißt das nicht, dass dieser auch wirklich verfügbar ist, da der Speicher von Nutz- und Metadaten gleichermaßen genutzt wird. Mögliche Deduplizierung und Kompression sorgen ebenfalls für ungenaue Anzeigen durch das Betriebssystem. Als Beispiel kopieren wir einmal den Inhalt des */usr*-Verzeichnisses (1,1 GB) in das gerade angelegte Filesystem:

```
# du -hs /usr
1.1G   /usr
```

```
# cp -r /usr /mnt/btrfs
# df -h /mnt/btrfs
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb         2.0G  552M  1.3G  30% /mnt/btrfs
```

Listing 4.26 Speicherauslastung von Btrfs

Um Verwirrung zu vermeiden, stellt Btrfs das `filesystem`-Subkommando zur Verfügung. Mit `show` können Sie sich den Aufbau anzeigen lassen:

```
# btrfs filesystem show /mnt/btrfs
Label: none  uuid: 71c1d2cc-479a-4090-ae7f-482d9ac21c29
    Total devices 2 FS bytes used 535.89MiB
    devid   1 size 2.00GiB used 853.50MiB path /dev/sdb
    devid   2 size 2.00GiB used 833.50MiB path /dev/sdc
```

Listing 4.27 Den Aufbau des Filesystems anzeigen lassen

Das `Diskfree`-Kommando in Listing 4.26 zeigt Ihnen 552 MB verbrauchten Speicherplatz. Das Kommando `btrfs filesystem show` aus Listing 4.27 gibt hingegen aus, dass 853 MiB belegt sind.

Wie der Speicher verwendet wird, zeigt Ihnen `btrfs filesystem df`. Wie Sie sehen können, werden 204 MiB für Metadaten verwendet:

```
# btrfs filesystem df /mnt/btrfs | column -ts,
Data          RAID1: total=620.75MiB  used=500.80MiB
Data          single: total=8.00MiB  used=0.00B
System        RAID1: total=8.00MiB  used=16.00KiB
System        single: total=4.00MiB  used=0.00B
Metadata      RAID1: total=204.75MiB  used=35.08MiB
Metadata      single: total=8.00MiB  used=0.00B
GlobalReserve single: total=16.00MiB  used=0.00B
```

Listing 4.28 »df«-Subkommando von »btrfs filesystem«

Eine genauere und auch längere Ausgabe bekommen Sie mit `btrfs filesystem usage`:

```
# btrfs filesystem usage /mnt/btrfs
Overall:
  Device size:                4.00GiB
  Device allocated:           1.65GiB
  Device unallocated:         2.35GiB
  Device missing:              0.00B
  Used:                        1.05GiB
  Free (estimated):           1.30GiB   (min: 1.30GiB)
```

```
Data ratio:                1.99
Metadata ratio:           1.96
Global reserve:          16.00MiB    (used: 0.00B)
Data,single: Size:8.00MiB, Used:0.00B
/dev/sdb                8.00MiB

Data,RAID1: Size:620.75MiB, Used:500.80MiB
/dev/sdb                620.75MiB
/dev/sdc                620.75MiB

Metadata,single: Size:8.00MiB, Used:0.00B
/dev/sdb                8.00MiB

Metadata,RAID1: Size:204.75MiB, Used:35.08MiB
/dev/sdb                204.75MiB
/dev/sdc                204.75MiB

System,single: Size:4.00MiB, Used:0.00B
/dev/sdb                4.00MiB

System,RAID1: Size:8.00MiB, Used:16.00KiB
/dev/sdb                8.00MiB
/dev/sdc                8.00MiB

Unallocated:
/dev/sdb                1.17GiB
/dev/sdc                1.19GiB
```

Listing 4.29 Das »usage«-Subkommando von »btrfs filesystem«

4.3 Fazit

Sie haben nun eine Reihe von Dateisystemen kennengelernt und erfahren, dass jedes seine eigenen Stärken und Schwächen hat. Das versetzt Sie in die Lage, für jede Aufgabe das passende Dateisystem zu wählen. Obwohl die Standarddateisysteme wie Ext4 und XFS prinzipiell für alle Einsatzzwecke geeignet sind, können Sie durch den Einsatz eines spezielleren Dateisystems nicht selten ein Plus an Geschwindigkeit oder Sicherheit herausholen.

Kapitel 5

Berechtigungen

Das Thema Berechtigungen nimmt einen großen Bereich in der System- und Netzwerkverwaltung ein. Hier geht es nicht nur um Dateisystemrechte, sondern auch um den Einsatz von ACLs im Dateisystem, von Quotas bis hin zum Einsatz von PAM und zur Beschränkung von Systemressourcen mit »ulimit«. Wenn Sie auch diese Punkte mit in Ihre Planung einfließen lassen, können Sie die Sicherheit in Ihrer Umgebung verbessern.

Eine der am meisten unterschätzten Planungen in größeren Umgebungen sind die Zugriffsrechte. Sie sollten sich schon, bevor Sie die Partitionierung Ihrer Datenträger planen, Gedanken über die zukünftige Rechtestruktur machen. Die Zeit, die Sie an dieser Stelle investieren, können Sie später einsparen, denn dann brauchen Sie keine umfassenden Änderungen mehr vorzunehmen.

Bei der Planung der Berechtigungen geht es nicht nur um die Vergabe der Dateisystemrechte, sondern auch um die Planung von Benutzern und Gruppen und um die zukünftige Struktur des Dateisystembaums. Auch sollten Sie den Einsatz von Dateisystem-ACLs in Betracht ziehen, besonders dann, wenn Sie Samba in Ihrer Umgebung verwenden. Auch die Verwendung der Spezialbits bei den Dateisystemrechten (wie des *Set Group ID*-(SGID)-Bits) sollten Sie planen. Ein weiterer wichtiger Punkt ist der Einsatz von Quotas im Dateisystem, um den ungehemmten Zuwachs der Daten einzugrenzen und damit Sie eventuell eine Reserve hinsichtlich des Speicherplatzes haben. Wie Sie sehen, gibt es auch in diesem Bereich eine Menge an Planungsarbeit.

5.1 User, Gruppen und Dateisystemstrukturen

Um Ihnen einen besseren Überblick über die Vergabe der Berechtigungen zu geben, haben wir in Tabelle 5.1 eine Übersicht über die Benutzer und Gruppen zusammengestellt, die in den darauffolgenden Übungen Verwendung finden.

Ein wichtiger Punkt, über den Sie sich vor dem Anlegen der Benutzer und Gruppen Gedanken machen sollten, ist die Wahl eines Namensstandards. Gerade in größeren Umgebungen wird die Verwaltung der Benutzer und Gruppen dadurch sehr viel übersichtlicher. In Tabelle 5.1 erkennen Sie diesen Namensstandard für die Benutzer und Gruppen: Bei den Gruppen

für spezielle Tätigkeiten werden immer vier Buchstaben der Abteilung mit zwei Buchstaben der Aufgabe verbunden. Der Benutzername setzt sich aus dem ersten Buchstaben des Vornamens und dem vollständigen Nachnamen zusammen. Für den Fall, dass in Ihrem Unternehmen mehrere Mitarbeiter denselben Namen haben, sollten Sie sich überlegen, wie Sie damit verfahren. Eine Lösung wäre, den zweiten und dritten Buchstaben des Vornamens zu verwenden.

Gruppe	Abteilung	Bereich	Mitglieder
benutzer	–	–	Default-Gruppe aller Mitarbeiter
Buchhaltung	Buchhaltung	–	alle Mitarbeiter der Buchhaltung
Produktion	Produktion	–	alle Mitarbeiter der Produktion
Verwaltung	Verwaltung	–	alle Mitarbeiter der Verwaltung
Alle	alle	alle	alle Mitarbeiter
buch-ig	Buchhaltung	Lohn und Gehalt	ptau, sdampf
buch-ps	Buchhaltung	Personal	ssorglos
prod-ak	Produktion	Arbeitskontrolle	dchecker, pwichtig
prod-av	Produktion	Arbeitsvorbereitung	chuber
prod-fg	Produktion	Fertigung	tknuf, kpfusch
prod-qs	Produktion	Qualitätssicherung	upruef, herbse
verw-gl	Verwaltung	Geschäftsleitung	bboss, sboss
verw-mk	Verwaltung	Marketing	wfutzi, plustig
verw-vt	Verwaltung	Vertrieb	kjeder, mkeiner
verw-ek	Verwaltung	Einkauf	ekauf, graus
verw-al	Verwaltung	Abteilungsleiter	plustig, dchecker, bboss, sboss
prod-al	Produktion	Abteilungsleiter	chuber, herbse, bboss, sboss
buch-al	Buchhaltung	Abteilungsleiter	ptau, bbos, sboss

Tabelle 5.1 Zuordnung der Benutzer und Gruppen



Wenn Sie verschiedene Clientsysteme wie Linux und Windows einsetzen, sollten Sie darauf achten, dass Sie keine Umlaute in den Benutzernamen verwenden und alle Buchstaben

des Benutzernamens kleinschreiben. Dann kann es auch bei der Verwendung verschiedener Zeichensätze bei der Anmeldung nicht zu Fehlern durch eine falsche Schreibweise kommen.

Wenn Sie ein neues Konzept für die Benutzerverwaltung einrichten wollen, sollten Sie verschiedene Dinge beachten. Versuchen Sie immer, Benutzer zu sinnvollen Gruppen zusammenzufassen und dann erst den Gruppen Berechtigungen zu geben. Vermeiden Sie es, Berechtigungen über *others* zu vergeben, denn so kann es passieren, dass plötzlich Benutzer Rechte an Dateien erhalten, die sie nicht haben sollen. Vermeiden Sie es, Rechte an einzelne Benutzer zu vergeben. Besser ist es, immer eine Gruppe einzurichten, denn dann ist es für Sie einfacher, einem anderen Benutzer dieselben Rechte zu geben.

Erst wenn Sie sich einen Plan bezüglich der Benutzer und Gruppen gemacht haben, sollten Sie sich Gedanken über den Aufbau einer Verzeichnisstruktur machen. Hier kann auch die Planung der Partitionierung stattfinden und festgelegt werden, welche Dateisysteme von welchen Servern gemountet werden. Eine gut durchdachte Dateisystemstruktur erspart Ihnen später umfangreiche und aufwendige Umstrukturierungen. Versuchen Sie immer, die Verzeichnisstruktur in einer Pyramidenform zu verwalten. Das heißt, die Struktur sollte auf der obersten Ebene wenige Verzeichnisse haben und nach unten hin immer breiter werden. In Listing 5.1 sehen Sie ein Beispiel mit dem Kommando `tree`:

```
adminbuch:/# tree daten/
daten/
|-- Abteilung
|   |-- Buchhaltung
|   |   |-- Lohnbuchhaltung
|   |   `-- Personalbuchhaltung
|   |-- Produktion
|   |   |-- Arbeitskontrolle
|   |   |-- Arbeitsvorbereitung
|   |   |-- Fertigung
|   |   `-- Qualitaetssicherung
|   `-- Verwaltung
|       |-- Einkauf
|       |-- Geschaeftsleitung
|       |-- Marketing
|       `-- Vertrieb
`-- Alle
```

Listing 5.1 Übersicht über die Verzeichnisstruktur mit »tree«

Aufgrund der geplanten Verzeichnisstruktur können Sie jetzt die Gruppen nach Ihrem vorher erstellten Namensstandard anlegen. Die Gruppen benötigen Sie ja, um effektiv Berechtigungen vergeben zu können. In Listing 5.2 sehen Sie ein Beispiel, wie eine Gruppenstruktur aussehen könnte, die auf die Dateisystemstruktur Bezug nimmt:

```
adminbuch:/daten# getent group
root:x:0:
daemon:x:1:
bin:x:2:
[...]
benutzer:*:1000:
Buchhaltung:*:10000:
Produktion:*:10001:
Verwaltung:*:10002:
Alle:*:10003:
buch-lg:*:10004:
buch-ps:*:10005:
prod-ak:*:10006:
prod-av:*:10007:
prod-fg:*:10008:
prod-qs:*:10009:
verw-gl:*:10010:
verw-mk:*:10011:
verw-vt:*:10012:
verw-ek:*:10013:
```

Listing 5.2 Gruppenplan mit »getent group«

Die Liste ist um einige Systemgruppen gekürzt dargestellt. Hier können Sie sehen, dass es zu jeder Abteilung eine Gruppe gibt, die den vollen Namen der Abteilung trägt. Die untergeordneten Gruppen lassen sich über Abkürzungen genau der entsprechenden Abteilung zuordnen.

Bei den Benutzernamen ist das so nicht möglich, da Sie sonst einen Benutzer, der die Abteilung wechselt, immer umbenennen müssten. Gerade aus diesem Grund ist die Planung der Gruppennamen so wichtig. So können Sie sehr schnell feststellen, in welcher Abteilung und in welchem Bereich einer Abteilung ein Mitarbeiter tätig ist.

5.2 Dateisystemberechtigungen

An dieser Stelle geht es nicht mehr um die Art und Weise, wie Dateisystemrechte vergeben werden, sondern um die Planung von Berechtigungsstrukturen. Da bei Linux-Systemen immer nur der Besitzer und der Benutzer `root` Rechte an Dateien und Verzeichnisse vergeben können, sollten Sie immer darauf achten, dass bei allen Verzeichnissen, die Sie als Systemverwalter den Benutzern zur Verfügung stellen, immer der Benutzer `root` als Besitzer eingetragen ist. So stellen Sie sicher, dass kein anderer Benutzer die Berechtigungen an den von Ihnen bereitgestellten Verzeichnissen ändern kann.

Nachdem Sie die Verzeichnisstruktur erzeugt und alle Gruppen angelegt haben, können Sie die Grundstruktur der Rechte einrichten. In Listing 5.3 sehen Sie die Rechtestruktur mit tree:

```
adminbuch:/# tree -pug daten/
daten/
|-- [drwxr-xr-x root    root    ] Abteilung
|  |-- [drwxr-x--- root    Buchhaltung] Buchhaltung
|  |  |-- [drwxrwx--- root    buch-lg ] Lohnbuchhaltung
|  |  `-- [drwxrwx--- root    buch-ps ] Personalbuchhaltung
|  |-- [drwxr-x--- root    Produktion] Produktion
|  |  |-- [drwxrwx--- root    prod-ak ] Arbeitskontrolle
|  |  |-- [drwxrwx--- root    prod-av ] Arbeitsvorbereitung
|  |  |-- [drwxrwx--- root    prod-fg ] Fertigung
|  |  `-- [drwxrwx--- root    prod-qs ] Qualitaetssicherung
|  `-- [drwxr-x--- root    Verwaltung] Verwaltung
|      |-- [drwxrwx--- root    verw-ek ] Einkauf
|      |-- [drwxrwx--- root    verw-gl ] Geschaeftsleitung
|      |-- [drwxrwx--- root    verw-mk ] Marketing
|      `-- [drwxrwx--- root    verw-vt ] Vertrieb
`-- [drwxrwx--- root    benutzer] Alle
```

Listing 5.3 Grundstruktur der Berechtigungen mit »tree«

Wie Sie in Listing 5.3 sehen können, wurde hier allen Verzeichnissen eine Gruppe zugeordnet, und die Rechte wurden so angepasst, dass nur noch die Gruppen Zugriff auf die Verzeichnisse haben. Bei den Verzeichnissen Buchhaltung, Produktion und Verwaltung handelt es sich nur um Verzeichnisse, die zur Strukturierung des Verzeichnisbaums dienen; hier haben deshalb die Benutzer keine Schreibrechte. Erst in den Verzeichnissen der Fachbereiche können die Benutzer Daten speichern.

Damit die Mitarbeiter der einzelnen Fachbereiche einer Abteilung Daten austauschen können, könnten Sie zusätzlich im Abteilungsverzeichnis ein Verzeichnis anlegen, das zum Datenaustausch dient, und dort anschließend der entsprechenden Abteilungsgruppe das Schreibrecht geben. Hier wäre es das Verzeichnis *Alle*.

Aber was tun Sie, wenn Sie an einem Verzeichnis unterschiedliche Rechte für unterschiedliche Gruppen vergeben wollen? Dies können Sie nicht mit den einfachen Dateisystemrechten abbilden. Dazu brauchen Sie die Dateisystem-ACLs aus Abschnitt 5.3.

5.2.1 Spezialbits

Um die Berechtigungen auch auf die durch Benutzer neu erstellten Ordner und Dateien übertragen zu können, sollten die Rechte um die Spezialbits erweitert werden. Zuvor jedoch noch eine Erklärung der einzelnen Bits:

Das SUID-Bit

Das SUID-Bit dient dazu, einem Benutzer Rechte an einer Datei zu geben, die er sonst nicht hat. Ein gutes Beispiel für dieses Bit ist die Berechtigung auf die Datei `/etc/shadow`, in der sich die Passwörter der Benutzer befinden. Die Berechtigung an dieser Datei ist standardmäßig wie in Listing 5.4 gesetzt:

```
adminbuch:/daten# ls -l /etc/shadow
-rw-r----- 1 root shadow 767 14. Mai 15:23 /etc/shadow
```

Listing 5.4 Rechte an der Datei »shadow«

Da kein Benutzer Mitglied der Gruppe `shadow` ist und nur der Benutzer `root` Schreibrechte an der Datei hat, muss den Benutzern die Änderung der Passwörter also über einen anderen Weg ermöglicht werden. Dazu sehen Sie in Listing 5.5 die Berechtigungen des Programms `/usr/bin/passwd`:

```
adminbuch:/daten# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 31704 14. Mai 2016 /usr/bin/passwd
```

Listing 5.5 Berechtigungen des Programms »/usr/bin/passwd«

Sie sehen hier, dass an der Stelle, an der normalerweise das »x« beim Benutzer gesetzt ist, ein »s« steht. Dieses »s« zeigt an, dass das SUID-Bit gesetzt ist. Ein kleines »s« zeigt dabei an, dass das »x« zusätzlich gesetzt ist. Würde an der Stelle ein großes »S« stehen, wäre das »x« nicht gesetzt. Dieses »s« sorgt dafür, dass der Benutzer, der das Programm startet, für die Laufzeit des Programms eine zusätzliche UID erhält, und zwar die UID des Eigentümers des Programms, in diesem Fall die UID des Benutzers `root`. Dadurch kann ein »normaler« Benutzer sein Passwort in der Datei `/etc/shadow` ändern.

Das SUID-Bit setzen Sie mit dem Kommando `chmod u+s <Programmname>`. Setzen Sie das SUID-Bit nur für ausführbare Programme. Bei anderen Dateien wäre es sinnlos.

Das SGID-Bit

Mit dem SGID-Bit können Sie die Verwaltung der Rechtestruktur in Ihrem Dateisystem beeinflussen. Normalerweise gibt es bei Linux keine Vererbung der Berechtigungen im Dateisystem, aber durch den Einsatz des SGID-Bits können Sie das in einem bestimmten Rahmen ändern. Wenn Sie an einem Verzeichnis das SGID-Bit setzen, wird ab diesem Zeitpunkt jeder neue Eintrag unterhalb des Verzeichnisses immer der Gruppe gehören, die in diesem Verzeichnis als besitzende Gruppe eingetragen ist.

Das SGID-Bit vererbt sich auch auf alle neu erstellten Unterverzeichnisse, die nach dem Setzen des SGID-Bits erzeugt werden. Wenn Sie auf Ihrer Verzeichnisstruktur das SGID-Bit an den einzelnen Bereichsverzeichnissen setzen, gehören anschließend alle Einträge der entsprechenden Gruppe, ohne dass ein Benutzer seine Standardgruppe ändern müsste.

Das SGID-Bit setzen Sie mit dem Kommando `chmod g+s <Verzeichnis>`. Das SGID-Bit macht nur Sinn, wenn es auf Verzeichnisse gesetzt wird. Nach der Änderung der Berechtigung in unserem Beispieldateisystembaum sieht dieser so wie in Listing 5.6 aus:

```
adminbuch:/# tree -pug daten/
daten/
|-- [drwxr-xr-x root  root  ] Abteilung
|  |-- [drwxr-x--- root  Buchhalt] Buchhaltung
|  |  |-- [drwxrws--- root  buch-lg ] Lohnbuchhaltung
|  |  `-- [drwxrws--- root  buch-ps ] Personalbuchhaltung
|  |-- [drwxr-x--- root  Produktion] Produktion
|  |  |-- [drwxrws--- root  prod-ak ] Arbeitskontrolle
|  |  |-- [drwxrws--- root  prod-av ] Arbeitsvorbereitung
|  |  |-- [drwxrws--- root  prod-fg ] Fertigung
|  |  `-- [drwxrws--- root  prod-qs ] Qualitaetssicherung
|  `-- [drwxr-x--- root  Verwaltung] Verwaltung
|      |-- [drwxrws--- root  verw-ek ] Einkauf
|      |-- [drwxrws--- root  verw-gl ] Geschaeftsleitung
|      |-- [drwxrws--- root  verw-mk ] Marketing
|      `-- [drwxrws--- root  verw-vt ] Vertrieb
`-- [drwxrws--- root  benutzer] Alle
```

Listing 5.6 Dateisystem mit gesetztem SGID-Bit

Wenn Sie noch die *umask* aller Benutzer auf den Wert 007 setzen, dann haben alle Dateien und Verzeichnisse später die passenden Rechte und die Verzeichnisse gehören auch immer den entsprechenden Gruppen.

Das Sticky-Bit

Über das Sticky-Bit können Sie dafür sorgen, dass nur der Besitzer einer Datei diese auch löschen kann, selbst wenn andere Benutzer aufgrund der Dateisystemberechtigungen das Recht hätten, die Datei zu löschen. Das Bit macht immer dann Sinn, wenn viele oder sogar alle Benutzer eines Systems auf ein Verzeichnis Zugriff haben und dort Dateien erstellen können.

So macht das System auch Gebrauch von dem Sticky-Bit, und zwar beim Verzeichnis */tmp*. Dort haben alle Benutzer immer das Schreibrecht, aber durch das Sticky-Bit kann nur der Besitzer einer Datei Einträge im Verzeichnis */tmp* löschen. Das Sticky-Bit setzen Sie mit dem Kommando `chmod o+t <Verzeichnis>`.

Im Beispiel wird jetzt das Sticky-Bit auf das Verzeichnis */daten/Alle* gesetzt, da dort alle Mitarbeiter des Unternehmens Schreibrecht haben. In Listing 5.7 sehen Sie die Ausgabe des Kommandos `tree`:

```

adminbuch:/# tree -pug daten/
daten/
|-- [drwxr-xr-x root  root  ] Abteilung
|   |-- [drwxr-x--- root  Buchhalt] Buchhaltung
|   |   |-- [drwxrws--- root  buch-lg ] Lohnbuchhaltung
|   |   `-- [drwxrws--- root  buch-ps ] Personalbuchhaltung
|   |-- [drwxr-x--- root  Produktion] Produktion
|   |   |-- [drwxrws--- root  prod-ak ] Arbeitskontrolle
|   |   |-- [drwxrws--- root  prod-av ] Arbeitsvorbereitung
|   |   |-- [drwxrws--- root  prod-fg ] Fertigung
|   |   `-- [drwxrws--- root  prod-qs ] Qualitaetssicherung
|   `-- [drwxr-x--- root  Verwaltung] Verwaltung
|       |-- [drwxrws--- root  verw-ek ] Einkauf
|       |-- [drwxrws--- root  verw-gl ] Geschaeftsleitung
|       |-- [drwxrws--- root  verw-mk ] Marketing
|       `-- [drwxrws--- root  verw-vt ] Vertrieb
`-- [drwxrws--T root  benutzer] Alle

```

Listing 5.7 Gesetztes Sticky-Bit am Verzeichnis »/daten/Alle«

Wie Sie hier sehen, steht an der Stelle, an der sonst unter *others* das »x« steht, ein großes »T«. Das zeigt an, dass unter dem »T« das »x« nicht gesetzt ist. Jetzt ist die Verzeichnisstruktur schon gut gegen unerlaubte Zugriffe gesichert, und die Rechte werden auch schon automatisch den entsprechenden Gruppen zugeordnet.

5.3 Erweiterte POSIX-ACLs

In diesem Abschnitt geht es darum, die Dateisystemrechte zu erweitern. Mithilfe der ACLs (*Access Control Lists*) ist es möglich, dass mehrere Gruppen oder Benutzer an einer Datei oder einem Verzeichnis unterschiedliche Rechte erhalten. Damit Sie die ACLs nutzen können, müssen Sie als Erstes das Dateisystem für die ACL-Unterstützung anpassen. Alle gängigen Dateisysteme wie *ext2*, *ext3*, *ext4*, *reiserfs*, *ifs* und *BtrFS* unterstützen ACLs. Aber die ACLs müssen beim Mounten des Dateisystems eventuell als Option mit angegeben werden. Sie können die entsprechende Option natürlich auch direkt in dem Eintrag der */etc/fstab* mit angeben. So werden die ACLs auch bei einem Systemstart sofort wieder aktiviert.



In aktuellen Distributionen nicht nötig

Bei allen in dieser Auflage vorgestellten Distributionen müssen Sie die ACLs nicht zum Mounten in die Datei */etc/fstab* eintragen, denn diese Option ist bereits in allen Dateisystemen fest eingebunden. Das Gleiche gilt für die erweiterten Dateisystemattribute. Prüfen Sie ein-

fach mit `getfacl`, ob Ihr verwendetes Dateisystem Ihnen ACLs anzeigt (auch wenn Sie noch keine ACLs gesetzt haben, bekommen Sie dann eine Ausgabe). Wenn das Kommando keine Fehlermeldung auswirft, brauchen Sie `acl` nicht als Mounthoption in der *fstab* anzugeben. Die entsprechenden Schritte können Sie dann überspringen.

In Listing 5.8 sehen Sie einen Auszug aus der Datei */etc/fstab* für ein Dateisystem mit ACLs:


```
/dev/sdb1      /daten          ext3    errors=remount-ro,acl 0      0
```

Listing 5.8 Dateisystemeintrag in der »*fstab*« mit aktivierten ACLs

Nachdem Sie den Eintrag in der Datei *fstab* angepasst haben, müssen Sie die Partition mit dem Kommando `mount -o remount /daten` remounten. Im Anschluss daran können Sie mit dem Kommando `mount` prüfen, ob die Option aktiv ist, wie in Listing 5.9 dargestellt:

```
adminbuch:/daten# mount
/dev/sda1 on / type ext3 (rw,errors=remount-ro)
/dev/sdb1 on /daten type ext3 (rw,errors=remount-rw,acl)
```

Listing 5.9 Gemountetes Dateisystem mit ACL-Unterstützung

Bei openSUSE ist die Option `acl` bereits nach der Installation des Systems für alle Datenpartitionen aktiv. Bei den aktuellen Debian- und Ubuntu-Versionen ist die Option `acl` fest im Dateisystem eingebunden. Hier müssen Sie die *fstab* nicht mehr anpassen. 

Nun ist das Dateisystem für den Einsatz von ACLs vorbereitet. Für die Verwaltung der ACLs stehen Ihnen zwei Kommandos zur Verfügung: Zum einen das Kommando `getfacl`, um sich die ACLs anzeigen zu lassen, und zum anderen das Kommando `setfacl`, um die ACLs zu setzen. Für diese beiden Kommandos müssen Sie das Paket `acl` auf Ihrem System installiert haben. Bei openSUSE ist es bereits installiert, bei Debian und Ubuntu müssen Sie das Paket nachinstallieren.

5.3.1 Setzen und Anzeigen von einfachen ACLs

Für alle Abteilungen soll es eine Gruppe von Abteilungsleitern geben, die auf alle Fachbereiche Zugriff erhalten sollen. Dafür werden drei neue Gruppen benötigt, in denen die Abteilungsleiter Mitglied werden. In Listing 5.10 sehen Sie wieder einen Auszug aus */etc/group*:

```
verw-al:*:10014:
prod-al:*:10015:
buch-al:*:10016:
Geschaeftsleitung:*:10017:
```

Listing 5.10 Auszug aus der Datei »*/etc/group*«

Um später der Geschäftsleitung noch zusätzliche Rechte geben zu können, gibt es jetzt auch eine neue Gruppe nur für die Mitarbeiter der Geschäftsleitung.

Jetzt werden als Erstes für alle Abteilungsleiter die ACLs in den Verzeichnissen der Fachbereiche gesetzt. In Listing 5.11 sehen Sie die Kommandos dazu:

```
adminbuch:/daten/Abteilung# setfacl -m g:verw-al:rwx Verwaltung/*
adminbuch:/daten/Abteilung# setfacl -m g:prod-al:rwx Produktion/*
adminbuch:/daten/Abteilung# setfacl -m g:buch-al:rwx Buchhaltung/*
```

Listing 5.11 Setzen der Rechte für die Abteilungsleiter

Zum Kommando `setfacl` gehören folgende Optionen und Argumente:

- ▶ `-m`
Die Option `-m` zeigt an, dass die ACL modifiziert werden soll. Soll eine ACL entfernt werden, muss die Option durch `-x` ersetzt werden.
- ▶ `g:buch-al:rwx`
Das `g` steht dafür, dass hier eine Gruppe zusätzlich in die ACL eingetragen werden soll. Wollen Sie über die ACL einen Benutzer mit zusätzlichen Rechten ausstatten, müssen Sie ein `u` einsetzen. Nach dem ersten Doppelpunkt folgt die Gruppe, die hier Rechte erhalten soll. Nach dem zweiten Doppelpunkt folgen die Rechte, die vergeben werden sollen.
- ▶ `Buchhaltung/*`
Das ist der Verzeichniseintrag, der die zusätzliche ACL erhalten soll.

In Listing 5.12 sehen Sie am Beispiel der Buchhaltung, dass die ACLs gesetzt sind:

```
adminbuch:/daten/Abteilung# ls -l Buchhaltung/
insgesamt 8
drwxrws---+ 2 root buch-lg 4096 26. Jul 12:03 Lohnbuchhaltung
drwxrws---+ 2 root buch-ps 4096 26. Jul 12:03 Personalbuchhaltung
```

Listing 5.12 Gesetzte ACLs für alle Unterverzeichnisse der Buchhaltung

An dem Pluszeichen hinter den Berechtigungen können Sie erkennen, dass dieses Verzeichnis mit ACLs ausgestattet ist. Um sich die ACLs genauer anzusehen, benötigen Sie das Kommando `getfacl`.

In Listing 5.13 sehen Sie die Ausgabe des Kommandos `getfacl`:

```
adminbuch:/daten/Abteilung# getfacl Buchhaltung/Lohnbuchhaltung/
# file: Buchhaltung/Lohnbuchhaltung/
# owner: root
# group: buch-lg
user::rwx
group::rwx
group:buch-al:rwx
```

```
mask::rwx
other::---
```

Listing 5.13 Ausgabe des Kommandos »getfacl«

Hier sehen Sie, dass zusätzlich zu der Gruppe buch-lg die Gruppe buch-al alle Rechte hat. Jetzt können die Abteilungsleiter auf alle Verzeichnisse ihrer Fachbereiche zugreifen.

Beim Auflisten der ACLs fällt auf, dass es dort einen Eintrag mask gibt. Die Maske kann dazu verwendet werden, bestimmte Rechte für alle Benutzer auszufiltern. Die genaue Verwendung der Maske wird in Abschnitt 5.3.3, »Setzen von erweiterten ACLs«, beschrieben.

Wenn jetzt ein Benutzer in einem der Verzeichnisse einen neuen Eintrag erstellt, fällt auf, dass die ACLs nicht mit übernommen werden. Dazu sehen Sie in Listing 5.14 ein Beispiel:

```
charly@adminbuch:/daten/Abteilung/Buchhaltung/Lohnbuchhaltung$ mkdir Briefe
charly@adminbuch:/daten/Abteilung/Buchhaltung/Lohnbuchhaltung$ ls -l
insgesamt 4
drwxrws--- 2 stefan buch-lg 4096 26. Jul 16:06 Briefe
```

Listing 5.14 Erstellung von Einträgen ohne Default-ACLs

Der Benutzer *Charly* hat hier ein neues Verzeichnis *Briefe* erzeugt, aber wie Sie an dem Ergebnis des Kommandos `ls -l` erkennen können, fehlt das Plussymbol nach den Berechtigungen. Im nächsten Schritt sollen deshalb die ACLs so gesetzt werden, dass diese immer für die Unterverzeichnisse übernommen werden.

5.3.2 Setzen von Default-ACLs

Im letzten Abschnitt fehlten die ACLs in einem neu erstellten Unterverzeichnis. Das werden wir nun beheben.

Mit dem Kommando `setfacl` haben Sie die Möglichkeit, *Default-ACLs* zu setzen. In Listing 5.15 werden die vorher gesetzten ACLs nun in Default-ACLs geändert:

```
adminbuch:/daten/Abteilung# setfacl -d -m g:buch-al:rwx Buchhaltung/*
adminbuch:/daten/Abteilung# getfacl Buchhaltung/Lohnbuchhaltung/
# file: Buchhaltung/Lohnbuchhaltung/
# owner: root
# group: buch-lg
user::rwx
group::rwx
group:buch-al:rwx
mask::rwx
other::---
default:user::rwx
```

```
default:group::rwx
default:group:buch-al:rwx
default:mask::rwx
default:other::---
```

Listing 5.15 Setzen und Anzeigen von Default-ACLs

Im ersten Schritt wurden die ACLs neu gesetzt, durch die Option `-d` werden diese ACLs zu Default-ACLs.

Im zweiten Schritt wurden die ACLs des Verzeichnisses *Lohnbuchhaltung* aufgelistet. Hier können Sie jetzt erkennen, dass vor allen ACLs der Begriff `default` steht. Jetzt werden diese ACLs auf alle neuen Unterverzeichnisse übernommen.



Beim Verwenden der Option `-d` ist die Reihenfolge der Optionen wichtig. Ändern Sie an der Stelle die Reihenfolge, kommt es zu einer Fehlermeldung.

Listing 5.16 zeigt den Unterschied beim erneuten Anlegen des Verzeichnisses *Briefe* durch den Benutzer *Charly*:

```
charly@adminbuch:/daten/Abteilung/Buchhaltung/Lohnbuchhaltung$ mkdir Briefe
```

```
charly@adminbuch:/daten/Abteilung/Buchhaltung/Lohnbuchhaltung$ ls -l Briefe
drwxrws---+ 2 charly buch-lg 4096 26. Jul 16:41 Briefe
```

```
charly@adminbuch:/daten/Abteilung/Buchhaltung/Lohnbuchhaltung$ getfacl Briefe/
# file: Briefe/
# owner: charly
# group: buch-lg
user::rwx
group::rwx
group:buch-al:rwx
mask::rwx
other::---
default:user::rwx
default:group::rwx
default:group:buch-al:rwx
default:mask::rwx
default:other::---
```

Listing 5.16 Anlegen eines neuen Verzeichnisses mit gesetzten Default-ACLs

Nachdem das neue Verzeichnis angelegt wurde, können Sie mit dem Kommando `ls -l Briefe` erkennen, dass das Pluszeichen wieder hinter den Berechtigungen steht. Ein Test mit `getfacl Briefe` zeigt, dass die ACLs genau wie beim übergeordneten Verzeichnis als Default-

ACLs gesetzt sind. Wenn Sie anschließend bei allen Fachbereichsverzeichnissen die ACLs auf diesem Wege setzen, werden immer alle ACLs auf neue Einträge im Dateisystem übernommen. Das gilt sowohl für Verzeichnisse als auch für Dateien.

5.3.3 Setzen von erweiterten ACLs

Nachdem Sie in den vorherigen Beispielen die Grundregeln für die ACLs kennengelernt haben, werfen wir an dieser Stelle einen etwas genaueren Blick auf die Maske. Die Maske dient dazu, Rechte gezielt für alle Einträge in der ACL zu filtern. Wenn Sie also ein bestimmtes Recht unbedingt verweigern wollen, können Sie das über die Maske festlegen. Die Maske sehen Sie, wenn Sie `getfacl` verwenden, als einen Eintrag in der Liste der ACLs.

Bei gesetzter ACL ändert »chmod« nur noch die Maske

Sobald Sie eine ACL auf einen Eintrag im Dateisystem gesetzt haben, wird bei dem Kommando `ls -l` vor dem Plussymbol nur noch die Maske angezeigt. Wenn Sie anschließend Berechtigungen mit `chmod` ändern, ändern Sie nur noch die Maske und nicht die Rechte. Wenn Sie ACLs verwenden, müssen Sie immer mit `setfacl` arbeiten.



In Listing 5.17 finden Sie ein Beispiel für dieses Verhalten:

```
adminbuch:/daten/Abteilung/Buchhaltung# chmod g-w Lohnbuchhaltung/
adminbuch:/daten/Abteilung/Buchhaltung# ls -l
insgesamt 8
drwxr-s---+ 3 root buch-lg 4096 26. Jul 16:41 Lohnbuchhaltung
drwxrws---+ 2 root buch-ps 4096 26. Jul 12:03 Personalbuchhaltung

adminbuch:/daten/Abteilung/Buchhaltung# getfacl Lohnbuchhaltung/
# file: Lohnbuchhaltung/
# owner: root
# group: buch-lg
user::rwx
group::rwx                #effective:r-x
group:buch-al:rwx         #effective:r-x
mask::r-x
other::---
default:user::rwx
default:group::rwx
default:group:buch-al:rwx
default:mask::rwx
default:other::---
```

Listing 5.17 Ändern der Maske mit »chmod«

Hier können Sie erkennen, dass zwar bei den Einträgen `group:` und `group:buch-al` noch die Rechte `rxw` eingetragen sind, aber hinter den Einträgen sehen Sie jetzt `#effective:r-x`. Dieser Eintrag zeigt die effektiven Rechte der Gruppen an, und das ist jetzt nur noch `r-x`.

Sie sehen auch, dass sich die Maske in `mask::r-x` geändert hat. Der Eintrag `default:mask::rxw` ändert sich nicht, da damit ja nur festgelegt wird, welche Berechtigung neue untergeordnete Einträge haben werden.

Setzen Sie die Maske mit `chmod` wieder zurück, verschwinden auch die `#effective:r-x`. Wollen Sie die Rechte der als Besitzer angezeigten Gruppe ändern, geht das nur noch über `setfacl`. Auch dazu sehen Sie ein Beispiel in Listing 5.18:

```
adminbuch:/daten/Abteilung/Buchhaltung# setfacl -m g::r-x Lohnbuchhaltung/
adminbuch:/daten/Abteilung/Buchhaltung# ls -l Lohnbuchhaltung/
drwxrws---+ 3 root buch-lg 4096 26. Jul 16:41 Lohnbuchhaltung/
```

```
adminbuch:/daten/Abteilung/Buchhaltung# getfacl Lohnbuchhaltung/
# file: Lohnbuchhaltung/
# owner: root
# group: buch-lg
user::rxw
group::r-x
group:buch-al:rxw
mask::rxw
other:---
default:user::rxw
default:group::rxw
default:group:buch-al:rxw
default:mask::rxw
default:other:---
```

Listing 5.18 Ändern der Gruppenrechte mit »setfacl«

In Listing 5.18 sieht es so aus, als ob nach dem Kommando `setfacl` keine Änderung an den Rechten stattgefunden hat, denn `ls -l` zeigt immer noch alle Rechte an. Aber denken Sie daran, dass hier nur noch die Maske angezeigt wird. Erst das Kommando `getfacl` zeigt die neuen Rechte für die besitzende Gruppe.

Wenn Sie schon eine bestehende Verzeichnisstruktur haben und für alle Verzeichnisse eine neue Default-ACL setzen, können Sie mit `setfacl` und der Option `-R` die Rechte auch rekursiv setzen.

Bevor Sie im nächsten Abschnitt sehen, wie ACLs gelöscht werden, folgt jetzt Tabelle 5.2, die Ihnen verdeutlichen soll, welche Rechte, Spezialbits und ACLs an den einzelnen Verzeichnissen vergeben wurden.

Verzeichnis	Besitzer	Gruppe	Rechte	ACL
daten	root	root	rwxI-XI-X	nein
Abteilung	root	root	rwxI-XI-X	nein
Buchhaltung	root	Buchhaltung	rwxI-X---	buch-al rwx
Lohnbuchhaltung	root	buch-lg	rwxIWS---	buch-al rwx
Personalbuchhaltung	root	buch-ps	rwxIWS---	buch-al rwx
Produktion	root	Produktion	rwxI-X---	prod-al rwx
Arbeitskontrolle	root	prod-ak	rwxIWS---	prod-al rwx
Arbeitsvorbereitung	root	prod-av	rwxIWS---	prod-al rwx
Fertigung	root	prod-fe	rwxIWS---	prod-al rwx
Qualitaetssicherung	root	verw-qs	rwxIWS---	prod-al rwx
Verwaltung	root	Verwaltung	rwxI-X---	verw-al rwx
Einkauf	root	verw-ek	rwxIWS---	verw-al rwx
Geschaeftsleitung	root	verw-gl	rwxIWS---	nein
Marketing	root	verw-mk	rwxIWS---	prod-al rwx
Vertrieb	root	verw-vt	rwxIWS---	prod-al rwx
alle	root	alle	rwxIWS--T	nein

Tabelle 5.2 Zuordnung der Berechtigungen zu den Verzeichnissen

5.3.4 Entfernen von ACLs

Wenn Sie irgendwann die ACLs von einem Eintrag wieder entfernen wollen, nutzen Sie dazu auch wieder das Kommando `setfacl`, wobei Sie hier zwischen den echten ACLs und den Default-ACLs unterscheiden müssen.

Im ersten Schritt sollen die Default-ACLs entfernt werden. Das sehen Sie in Listing 5.19:

```
adminbuch:/daten/Abteilung/Buchhaltung# setfacl -k Lohnbuchhaltung/
```

```
adminbuch:/daten/Abteilung/Buchhaltung# ls -l Lohnbuchhaltung/
drwxrws---+ 3 root buch-lg 4096 26. Jul 16:41 Lohnbuchhaltung/
```

```
adminbuch:/daten/Abteilung/Buchhaltung# getfacl Lohnbuchhaltung/  
# file: Lohnbuchhaltung/  
# owner: root  
# group: buch-lg  
user::rwx  
group::r-x  
group:buch-al:rwx  
mask::rwx  
other::---
```

Listing 5.19 Entfernen der Default-ACLs

Durch die Option `-k` werden alle Default-ACLs entfernt. Das Kommando gibt keine Meldung aus, auch wenn keine Default-ACLs gesetzt sind. Das Kommando `ls -l` zeigt jetzt zwar noch an, dass es ACLs gibt, aber das Kommando `getfacl` zeigt nur noch die ACLs, aber keine Default-ACLs mehr. Im nächsten Schritt sollen jetzt auch noch die ACLs entfernt werden, sodass nur noch die Standard-Dateisystemrechte übrig bleiben. Diesen Schritt sehen Sie in Listing 5.20:

```
adminbuch:/daten/Abteilung/Buchhaltung# setfacl -b Lohnbuchhaltung/  
adminbuch:/daten/Abteilung/Buchhaltung# ls -l Lohnbuchhaltung/  
drwxr-s--- 3 root buch-lg 4096 26. Jul 16:41 Lohnbuchhaltung/  
  
adminbuch:/daten/Abteilung/Buchhaltung# getfacl Lohnbuchhaltung/  
# file: Lohnbuchhaltung/  
# owner: root  
# group: buch-lg  
user::rwx  
group::r-x  
other::---
```

Listing 5.20 Entfernen der ACLs

Nach dem Kommando `setfacl` ist das Plussymbol hinter den Dateisystemrechten verschwunden, und das Kommando `getfacl` zeigt auch nur noch die Standardberechtigungen an. Noch etwas ist hier zu sehen: In Listing 5.17 wurde die Maske mit `chmod` geändert. Nach dem Entfernen der ACLs wird nun die Einstellung der Maske für die Gruppe als Gruppenberechtigung übernommen.

5.3.5 Sichern und Zurückspielen von ACLs

Ein Problem gibt es bei der Verwendung von Dateisystem-ACLs jedoch: Nicht alle Backupprogramme unterstützen das Sichern und Zurückspielen der ACLs. Sie können aber die ACLs vor jeder Datensicherung mit `getfacl -R /daten/Abteilung > acl.bak` rekursiv sichern und später immer mit `setfacl --restore=acl.bak` wiederherstellen.

5.4 Erweiterte Dateisystemattribute

Neben den ACLs gibt es noch die erweiterten Dateisystemattribute. Diese Attribute können Sie bei den Dateisystemen *ext2*, *ext3* und *ext4* verwenden.

Wenn Sie die Dateisystemattribute auf einer Partition einsetzen, die später über NFS Benutzern zur Verfügung gestellt werden soll, wirken die Attribute auf den Clients, können aber nicht angezeigt oder geändert werden.

Die erweiterten Dateisystemattribute können Sie zusätzlich zu den Dateisystemrechten verwenden. Die unterschiedlichen Aufgaben der einzelnen Attribute werden im Folgenden genauer erklärt. Damit Sie die Dateisystemattribute verwenden können, muss das Dateisystem mit der Option `user_xattr` gemountet sein. Diese Option können Sie genau wie schon die Option `acl` direkt in die Datei `/etc/fstab` eintragen.

Bei openSUSE ist diese Option schon bei allen Dateisystemen eingetragen, die während der Installation vorhanden waren. Bei Debian und Ubuntu ist die Option bereits fest im Dateisystem implementiert und muss auch für neue Dateisysteme nicht mehr in die *fstab* eingetragen werden.

Anschließend muss die Partition mit dem Kommando `mount -o remount /daten` neu gemountet werden. Danach wird die Änderung auch mit dem Kommando `mount` angezeigt. Die Attribute können jetzt mit dem Kommando `lsattr` aufgelistet und mit dem Kommando `chattr` geändert werden.

Für diese Kommandos wird das Paket `attr` benötigt. Bei openSUSE ist es bereits installiert, bei Debian und Ubuntu muss das Paket noch nachinstalliert werden.

Die Attribute werden mit dem Kommando `chattr` verändert. Wenn Sie vor einem Attribut ein Minus eingeben, wird das entsprechende Attribut entfernt; mit einem Plus wird das Attribut gesetzt. Verwenden Sie ein Gleichheitszeichen, wird nur das Attribut gesetzt, alle anderen möglicherweise bereits gesetzten Attribute werden zurückgesetzt.

Bei den Attributen gibt es einige, die von jedem Benutzer gesetzt werden können, und einige, die nur vom Benutzer `root` gesetzt werden dürfen.

5.4.1 Attribute, die jeder Benutzer ändern kann

- ▶ Das Attribut `d`
Durch Setzen dieses Attributs verhindern Sie, dass die entsprechende Datei bei einer Datensicherung mit dem Kommando `dump` mitgesichert wird.
- ▶ Das Attribut `s`
Wenn ein Benutzer dieses Attribut setzt, wird die Datei beim Löschen mit Nullen überschrieben.

► Das Attribut A

Wenn Sie das Attribut A setzen, wird die *atime* der Datei nicht verändert. Das kann die Performance beim Schreiben erhöhen. Besser ist es aber, hier die Option *noatime* für das gesamte Dateisystem in der Datei */etc/fstab* zu setzen.

5.4.2 Attribute, die nur »root« ändern kann

Das Attribut a

Wenn Sie als *root* dieses Attribut auf eine Datei setzen, kann der Inhalt der Datei nicht mehr geändert werden, es können nur noch Daten an die Datei angehängt werden. In Listing 5.21 sehen Sie dazu ein Beispiel:

```
adminbuch:/daten/Alle# touch datei1.txt
adminbuch:/daten/Alle# chattr +a datei1.txt
adminbuch:/daten/Alle# lsattr datei1.txt
-----a----- datei1.txt

adminbuch:/daten/Alle# echo "Eine neue Zeile" > datei1.txt
-bash: datei1.txt: Die Operation ist nicht erlaubt

adminbuch:/daten/Alle# echo "Eine neue Zeile anhängen " >> datei1.txt
```

Listing 5.21 Wirkung des Attributs »a«

Mit dem ersten Kommando wird einfach eine leere Datei erzeugt, bei der im zweiten Schritt das Attribut *a* gesetzt wird. Im Anschluss wird versucht, eine Zeile in die Datei mit einer einfachen Umleitung zu schreiben. Diese Operation wird aufgrund des Attributs verboten. Erst im nächsten Schritt wird eine Zeile an die Datei angehängt, und jetzt ist das Schreiben in die Datei möglich.

Das Attribut i

Durch das Attribut *i* wird die Datei immun gegen das Ändern, Umbenennen und Löschen. Auch *root* kann eine Datei mit diesem Attribut nicht löschen, ohne vorher das Attribut zurückzusetzen. Auch dazu sehen Sie ein Beispiel in Listing 5.22:

```
adminbuch:/daten/Alle# chattr +i datei1.txt
adminbuch:/daten/Alle# lsattr datei1.txt
-----i----- datei1.txt

adminbuch:/daten/Alle# echo "Eine zweite Zeile anhängen " >> datei1.txt
-bash: datei1.txt: Keine Berechtigung
```

```

adminbuch:/daten/Alle# ls -l datei1.txt
-rw-r--r-- 1 root stefan 27 27. Jul 18:02 datei1.txt

adminbuch:/daten/Alle# rm datei1.txt
rm: Entfernen von "datei1.txt" nicht möglich:\
  Die Operation ist nicht erlaubt
adminbuch:/daten/Alle# mv datei1.txt datei1a.txt
mv: Verschieben von "datei1.txt" nach "datei1a.txt" nicht möglich:\
  Die Operation ist nicht erlaubt

adminbuch:/daten/Alle# chattr -i datei1.txt
adminbuch:/daten/Alle# rm datei1.txt

```

Listing 5.22 Wirkung des Attributs »i«

Alle Aktionen wurden hier als Benutzer root durchgeführt. Wie Sie sehen, ist danach keine Aktion mit der Datei mehr möglich. Erst wenn das Attribut von der Datei entfernt wird, kann die Datei wieder gelöscht werden.

5.4.3 Weitere Attribute

Neben diesen Attributen gibt es noch weitere Attribute, die zum Teil experimentell oder nicht von so großer Bedeutung für den täglichen Gebrauch sind. Alle Attribute werden ausführlich in der Manpage zu `chattr` beschrieben. Wenn Sie die Attribute verwenden wollen, sollten Sie immer einen Blick auf diese Manpage werfen, um mögliche Komplikationen zu vermeiden, denn nicht alle Attribute funktionieren in allen Dateisystemen.

5.5 Quotas

In vielen Fällen ist es sinnvoll, den Platz im Dateisystem für Benutzer oder Gruppen zu beschränken, um die Datensammelwut und den dadurch entstehenden Verbrauch an Festplattenplatz etwas einzuschränken, besonders wenn es um die Heimatverzeichnisse und Profilverzeichnisse der Benutzer geht. Aber oft kann es auch sinnvoll sein, die Verzeichnisse von Abteilungen oder Projekten einzuschränken.

Damit Sie diese Beschränkungen einrichten können, gibt es unter Linux das Quota-System. Mit dem Quota-System können Sie den Festplattenplatz für Benutzer und Gruppen auf jeder Partition einschränken. Da sich die Beschränkung immer auf eine Partition bezieht, können Sie bei einer gut überlegten Partitionierung den Platz den Aufgaben entsprechend begrenzen. In den folgenden Beispielen soll die vorher eingerichtete Partition für die Daten zusätzlich mit User- und Group-Quotas ausgestattet werden.

5.5.1 Installation und Aktivierung der Quotas

Im ersten Schritt müssen Sie auf den verschiedenen Distributionen das Quota-System aktivieren. Die Distributionen gehen dabei wieder unterschiedliche Wege. Deshalb beschreiben wir hier die Installation und Aktivierung der Quotas für die verschiedenen Distributionen.

Debian und Ubuntu

Bei Debian und Ubuntu müssen Sie als Erstes die beiden Pakete `quota` und `quotatool` installieren. Anschließend müssen Sie die Datei `/etc/fstab` für das entsprechende Dateisystem so wie in Listing 5.23 anpassen:

```
/dev/sdb1 /daten ext3 usrquota,grpquota 0 0
```

Listing 5.23 Anpassung der »`/etc/fstab`« für die Verwendung von Quotas

Durch die Option `usrquota` werden die User-Quotas und durch die Option `grpquota` die Group-Quotas aktiviert. Im Anschluss daran müssen Sie das Dateisystem neu mounten, um die Option zu aktivieren. Im nächsten Schritt muss das Quota-System für den ersten Einsatz initialisiert werden. Das führen Sie mit dem Kommando `quotacheck` durch.



Da nach der Installation der Pakete das Quota-System sofort aktiviert wurde, müssen Sie vor der Initialisierung das Quota-System erst mit dem Kommando `quotaoff` abschalten. Erst dann kann das Quota-System initialisiert werden. Im Anschluss können Sie das Quota-System für die Partition `/daten` erneut aktivieren. In Listing 5.24 sehen Sie diesen Vorgang:

```
adminbuch:~# quotacheck -cug /daten/
quotacheck: Quota for users is enabled on mountpoint /daten so quotacheck\
           might damage the file.
Please turn quotas off or use -f to force checking.
```

```
adminbuch:~# /sbin/quotaoff /daten
adminbuch:~# /sbin/quotacheck -cug /daten/
adminbuch:~# quotaon /daten
```

Listing 5.24 Initialisierung des Quota-Systems

Beim ersten Versuch, das Quota-System zu initialisieren, kommt es zu der Warnung, dass das Quota-System bereits aktiv ist und es dadurch zu Problemen kommen kann. Deshalb wird im nächsten Schritt das Quota-System abgeschaltet. Der nächste Versuch, das Quota-System mit dem Kommando `quotacheck` zu initialisieren, wird anschließend ohne Warnung durchgeführt. Die Parameter des Programms haben dabei die folgenden Bedeutungen:

► **-c**

Die Quotas werden in jeder Partition in einer eigenen Datei abgespeichert, und zwar getrennt jeweils für die User-Quotas und die Group-Quotas. Diese Dateien befinden sich

immer in der obersten Ebene des Dateisystems, für das die Quotas eingerichtet wurden. Die Dateien heißen *aquota.user* und *aquota.group*. Durch die Option `-c` wird bei der Ausführung von `quotacheck` überprüft, ob die Dateien bereits vorhanden sind; wenn nicht, werden die Dateien erzeugt.

- ▶ `-u`
Diese Option aktiviert die User-Quotas.
- ▶ `-g`
Diese Option aktiviert die Group-Quotas.

Nach der Initialisierung können Sie das Quota-System mit dem Kommando `quotaon` wieder aktivieren. Ab jetzt ist das Quota-System für das entsprechende Dateisystem aktiv.

openSUSE

Bei openSUSE sind die entsprechenden Pakete bereits installiert. Dort müssen Sie nur, genau wie bei Debian und Ubuntu, die Datei */etc/fstab* für die User- und Group-Quotas anpassen. Die Initialisierung des Quota-Systems wird hier entweder beim nächsten Start des Servers durchgeführt oder von Hand mit `systemctl start systemd-quotacheck` gestartet.

Dabei wird auch geprüft, ob die Dateien für die Verwaltung der Quotas bereits vorhanden sind. Wenn nicht, werden die Dateien automatisch erstellt. Das System prüft bei jedem Neustart, ob es Dateisysteme in der Datei */etc/fstab* gibt, bei denen `quota` als Option eingetragen ist. Bei diesen Partitionen wird das Quota-System automatisch gestartet.

5.5.2 Journaling-Quotas

Für alle Journaling-Dateisysteme wie *ext3* kann auch für die Quotas ein Journal geführt werden. Dadurch wird eine Überprüfung der Quota-Einträge nach einem Systemabsturz überflüssig. Durch das Journal werden die Quotas, genau wie das Dateisystem, schneller und automatisch geprüft. Auch wird die Prüfung erheblich schneller durchgeführt, was gerade bei größeren Dateisystemen wichtig sein kann. Damit Sie die *Journaling-Quotas* verwenden können, muss der Kernel diese Möglichkeit unterstützen. Bei allen Distributionen ist dies der Fall. Das können Sie auch ganz einfach testen, indem Sie nach der Konfiguration, wie in Listing 5.25 zu sehen, einfach das Kommando `quotacheck` wie folgt aufrufen:

```
adminbuch:~# quotaoff /daten
adminbuch:~# quotacheck -vug /daten
quotacheck: Your kernel probably supports journaled quota but you are not using it.
```

Consider switching to journaled quota to avoid running quotacheck after an unclean shutdown.

Listing 5.25 Test, ob die Journaling-Quotas vom Kernel unterstützt werden

Hier sehen Sie, dass es bei der Überprüfung zu einer Warnmeldung kommt und dass `quotacheck` vorschlägt, die Journaling-Funktion zu aktivieren. Für alle Distributionen ist dieser Vorgang identisch, Sie müssen lediglich die Einträge in der Datei `/etc/fstab` ändern. Die Datei sieht so wie in Listing 5.26 aus:

```
/dev/sdb1 /daten ext3\  
errors=remount-ro,acl,user_xattr,usrquota,usrjquota=aquota.user,\  
grpquota,grpjquota=aquota.group,jqfmt=vfsv0 0 0
```

Listing 5.26 Einträge in der »fstab« für Journaling-Quotas

Zu den schon vorher eingetragenen Optionen kommen jetzt drei weitere Optionen hinzu:

- ▶ `usrjquota=aquota.user`
Hier wird das Journal für die User-Quotas festgelegt.
- ▶ `grpjquota=aquota.group`
Hier wird das Journal für die Group-Quotas festgelegt.
- ▶ `jqfmt=vfsv0`
Mit dieser Option wird das Format des Journals für die Quotas festgelegt. Ohne diese Option lässt sich die Partition nicht mehr mounten, da das Journal für die Quotas nicht geschrieben werden kann.



Wichtig ist auch, dass Sie auf jeden Fall die Optionen `usrquota` und `grpquota` in der Datei `/etc/fstab` stehen lassen, denn die neuen Optionen sind lediglich für das Journal verantwortlich und nicht für die Aktivierung der Quotas. Wenn Sie jetzt die Quotas auf dem Dateisystem testen, wird die Warnung nicht mehr erscheinen.

5.5.3 Quota-Einträge verwalten

Als Erstes sollten Sie prüfen, ob das Quota-System auch wirklich aktiv ist. Die Prüfung wird mit dem Kommando `quotaon` durchgeführt, wie Sie in Listing 5.27 sehen:

```
adminbuch:~# quotaon -p /daten  
group quota on /daten (/dev/sdb1) is on  
user quota on /daten (/dev/sdb1) is on
```

Listing 5.27 Überprüfung des Quota-Systems

Wie Sie sehen, ist das Quota-System auf der Partition `/daten` sowohl für User-Quotas als auch für Group-Quotas aktiv.

Wenn das Quota-System aktiv ist, können Sie die Beschränkungen für Benutzer und Gruppen eintragen. Dazu wird das Kommando `edquota` verwendet. Doch bevor wir das Setzen von Quotas für die Benutzer und Gruppen an einem Beispiel zeigen, sollen hier noch einige Begriffe in Bezug auf Quotas erklärt werden:

► **softlimit**

Mit dem Softlimit können Sie eine erste Grenze für einen Benutzer oder eine Gruppe setzen. Wenn diese Grenze erreicht ist, bekommt der Benutzer oder ein Mitglied der Gruppe eine Warnung, dass das Softlimit überschritten wurde. Wenn die Gnadenfrist (*grace period*) nicht gesetzt ist, passiert an der Stelle nichts weiter. Wenn aber eine *grace period* gesetzt wurde, darf das Softlimit nur so lange überschritten werden, wie die Zeit bei der *grace period* eingestellt ist. Wenn diese Zeit abgelaufen ist, kann der Benutzer keine Daten mehr speichern, auch wenn das Hardlimit noch nicht erreicht wurde. Der Wert wird in Blöcken zu 1.024 Byte angegeben, unabhängig von der eingestellten Blockgröße der Partition. Wenn Sie einem Benutzer 90 MB Platz als Softlimit zuweisen wollen, wären das somit $90 \text{ MB} \times 1.024 \text{ K} = 92.160$ Blöcke.

Es gibt auch die Möglichkeit, die Anzahl der *Inodes* zu begrenzen. Dann kann der Benutzer nur eine bestimmte Anzahl an *Inodes* verwenden.

► **hardlimit**

Mit dem Hardlimit legen Sie die absolute Obergrenze des Speicherplatzes eines Benutzers oder einer Gruppe fest. Über diesen Wert hinaus können keine Daten gespeichert werden, auch dann nicht, wenn eine gesetzte *grace period* noch nicht abgelaufen ist. Auch dieser Wert wird in Blöcken zu 1.024 Byte oder *Inodes* festgelegt. Der Wert für das Hardlimit muss immer größer sein als der Wert für das Softlimit.

► **grace period**

Bei diesem Wert handelt es sich um eine »Gnadenfrist« für den Benutzer. Solange diese Zeit nicht abgelaufen ist, kann er auch über das Softlimit hinaus Daten auf der Partition speichern, aber nur bis hin zum Hardlimit. Als Zeitangabe können Sie *seconds*, *minutes*, *hours*, *days*, *weeks* und *months* verwenden.

Wenn Sie die Quota-Einträge für einen Benutzer bearbeiten, startet nach dem Aufruf von `edquota <Benutzer>` der von Ihnen am System eingestellte Editor und zeigt Ihnen die momentan verwendeten Blöcke des Benutzers und die eingestellten Limits. Hier können Sie jetzt die Änderungen vornehmen. In Listing 5.28 sehen Sie die eingetragenen Quotas für einen Benutzer:

```
adminbuch:~# edquota -u charly
Disk quotas for user charly (uid 10000):
  Filesystem      blocks      soft      hard    inodes    soft    hard
  /dev/sdb1        4          92160    102400      2         0      0
```

Listing 5.28 Verwendung von »edquota« für einen Benutzer

Jetzt hat der Benutzer *Charly* nur noch den durch die Quotas eingeschränkten Plattenplatz zur Verfügung. Wollen Sie später den momentan verwendeten Speicherplatz prüfen, können Sie das mit dem Kommando `repquota` durchführen, so wie Sie es in Listing 5.29 sehen:

```

adminbuch:~# repquota -u /daten
*** Report for user quotas on device /dev/sdb1
Block grace time: 7days; Inode grace time: 7days
                Block limits                File limits
User           used  soft  hard  grace  used  soft  hard  grace
-----
root    --   35936     0     0           19     0     0
charly  --     4  92160 102400         2     0     0
    
```

Listing 5.29 Erstellen eines Reports mit »repquota« für alle Benutzer

Eine Überprüfung für einzelne Benutzer ist so nicht möglich, denn über die Option `-u` werden alle Quotas aller Benutzer angezeigt.

Im nächsten Schritt soll im direkten Anschluss für die Gruppe `benutzer` noch ein Quota-Eintrag erstellt werden. Das sehen Sie in Listing 5.30:

```

adminbuch:~# edquota -g benutzer
Disk quotas for group benutzer (gid 1000):
  Filesystem    blocks    soft    hard    inodes    soft    hard
  /dev/sdb1      4      9000   10000      1         0     0
    
```

Listing 5.30 Verwendung von »edquota« für eine Gruppe

Auch hier können die Quota-Einträge für Gruppen wieder mit dem Kommando `repquota` überprüft werden, wie Sie in Listing 5.31 sehen:

```

adminbuch:~# repquota -g /daten
*** Report for group quotas on device /dev/sdb1
Block grace time: 7days; Inode grace time: 7days
                Block limits                File limits
Group          used  soft  hard  grace  used  soft  hard  grace
-----
root    --   35880     0     0           5     0     0
stefan  --     4   9000 10000         1     0     0
Buchhaltung --   4     0     0         1     0     0
Produktion --   4     0     0         1     0     0
Verwaltung --   4     0     0         1     0     0
buch-lg  --   8     0     0         3     0     0
buch-ps  --   4     0     0         1     0     0
prod-ak  --   4     0     0         1     0     0
prod-av  --   4     0     0         1     0     0
prod-fg  --   4     0     0         1     0     0
prod-qs  --   4     0     0         1     0     0
verw-gl  --   4     0     0         1     0     0
verw-mk  --   4     0     0         1     0     0
    
```



```

verw-vt  --      4      0      0              1      0      0
verw-ek  --      4      0      0              1      0      0

```

Listing 5.31 Erstellen eines Reports mit »repquota« für alle Gruppen

Wie Sie sehen, werden hier auch wieder alle Gruppen aufgelistet, mit Ausnahme der Systemgruppen. Jetzt soll zum Abschluss noch mit dem Kommando `edquota` die *grace period* gesetzt werden. Das sehen Sie in Listing 5.32:

```

adminbuch:~# edquota -t
Grace period before enforcing soft limits for users:
Time units may be: days, hours, minutes, or seconds
  Filesystem          Block grace period      Inode grace period
  /dev/sdb1              1days                   4days

```

```

adminbuch:~# repquota /daten
*** Report for user quotas on device /dev/sdb1
Block grace time: 24:00; Inode grace time: 4days

```

User		Block limits			File limits				
		used	soft	hard	grace	used	soft	hard	grace
root	--	35936	0	0		19	0	0	
charly	--	4	92160	102400		2	0	0	

Listing 5.32 Setzen der Gnadenfrist mit »edquota«

Die Zeiten für Blocklimits wurden hier auf einen Tag gesetzt und die Zeiten für die Inode-Limits auf vier Tage. Im Anschluss daran können Sie mit `repquota /daten` die Werte sofort überprüfen.

Damit Sie nicht für jeden Benutzer einzeln die Quotas setzen müssen, können Sie die Quota-Einstellungen eines Benutzers oder einer Gruppe mit dem Kommando `edquota` auch kopieren. In Listing 5.33 sehen Sie den Ablauf:

```

adminbuch:~# edquota -p charly -u stefan
adminbuch:~# repquota -uv /daten
*** Report for user quotas on device /dev/sdb1
Block grace time: 24:00; Inode grace time: 4days

```

User		Block limits			File limits				
		used	soft	hard	grace	used	soft	hard	grace
root	--	35936	0	0		19	0	0	
stefan	--	0	92160	102400		0	0	0	
charly	--	4	92160	102400		2	0	0	

Listing 5.33 Kopieren eines Quota-Eintrags

So müssen Sie nicht jeden Benutzer einzeln editieren. Als Parameter hinter der Option `-u` können Sie auch mehrere Benutzer angeben. Von dem Zeitpunkt an kann jeder Benutzer nur noch den Speicherplatz verwenden, den Sie ihm zur Verfügung gestellt haben.

Bleibt noch die Frage: Was passiert, wenn auf einer Partition sowohl Group- als auch User-Quotas eingerichtet wurden und ein Benutzer Mitglied der Gruppe ist? Ganz einfach, die User-Quotas haben immer die höhere Priorität.

5.6 Pluggable Authentication Modules (PAM)

Ohne die *Pluggable Authentication Modules (PAM)* müsste jede Applikation, die eine Anmeldung benötigt, für jede Authentifizierungsmethode neu geschrieben werden. Wird eine Anwendung hingegen *pamifiziert*, braucht sich die Anwendung nicht mehr selbst um die Authentifizierung zu kümmern, denn das übernimmt jetzt PAM.

Bei einer Anmeldung nimmt dann PAM die Anmeldedaten des Benutzers entgegen und leitet diese Daten durch die entsprechenden Module an eine Benutzerdatenbank. Diese prüft die Daten und leitet das Ergebnis, wieder über PAM, zurück an die Anwendung.

Der Begriff *pamifiziert* bedeutet, dass das entsprechende Programm die PAM-Funktion verwenden kann. Ob ein Programm die PAM-Funktionen nutzen kann, können Sie testen, indem Sie überprüfen, ob das Programm gegen die PAM-Bibliotheken gelinkt wurde.

In Listing 5.34 sehen Sie als Beispiel das Programm `login`:

```
adminbuch:~# ldd /bin/login
linux-gate.so.1 => (0xb76ee000)
libpam.so.0 => /lib/libpam.so.0 (0xb76dc000)
libpam_misc.so.0 => /lib/libpam_misc.so.0 (0xb76d9000)
libc.so.6 => /lib/i686/cmov/libc.so.6 (0xb757d000)
libdl.so.2 => /lib/i686/cmov/libdl.so.2 (0xb7579000)
/lib/ld-linux.so.2 (0xb76ef000)
```

Listing 5.34 Test des Programms »login« auf PAM-Unterstützung

Hier sehen Sie, dass das Programm `login` gegen die beiden Bibliotheken `libpam.so.0` und `libpam_misc.so.0` gelinkt wurde und somit die PAM-Funktionen unterstützt. Neben dem Programm `login` gibt es eine große Anzahl an Programmen und Diensten, die PAM verwenden können. Einige davon sind zum Beispiel `su`, `sudo` und `xscreensaver`.

Da all diese Programme PAM unterstützen, muss bei PAM nur noch das entsprechende Modul eingebunden werden, das dann eine bestimmte Art der Authentifizierung unterstützt. Zur Authentifizierung über PAM können zum Beispiel Passwortdateien wie die `/etc/passwd`, LDAP, Fingerprints Scanner oder Smartcards verwendet werden. PAM ist aber nicht zuständig für die Bereitstellung von Benutzerinformationen, die nichts mit der Authentifizierung zu

tun haben, wie zum Beispiel das Zuordnen einer UID zum Benutzernamen bei der Vergabe von Rechten oder der Verwendung von `ls -l`. Das ist die Aufgabe des *Name Service Switch (NSS)*.

5.6.1 Verschiedene PAM-Typen

Wie zuvor schon erwähnt wurde, ist PAM modular aufgebaut, und die verschiedenen Module können von unterschiedlichen Programmierern oder Hardwareherstellern bereitgestellt werden. Alle Module können den verschiedenen Modultypen zugeordnet werden. Es gibt vier verschiedene Modultypen innerhalb des PAM-Systems:

► **auth**

Wenn Sie einem Modul diesen Modultyp zuweisen, kümmert sich das Modul um die Authentifizierung von Benutzern über eine Passwortabfrage. Aber auch biometrische Methoden, wie zum Beispiel ein Fingerprints Scanner, können hier eingebunden werden.

► **account**

Wenn Sie ein Modul hier eintragen, wird während der Authentifikation geprüft, ob es Restriktionen für den Benutzer gibt. Restriktionen können zum Beispiel Anmeldezeitbeschränkungen oder der Ablauf des Passworts sein.

► **password**

Hier geht es darum, was zu beachten ist, wenn ein Benutzer ein Passwort ändert. Es kann zum Beispiel das Modul *pam_cracklib* eingebunden werden, um Passwörter auf Komplexität zu prüfen.

► **session**

Hier können Sie festlegen, was passieren soll, wenn sich ein Benutzer angemeldet hat. Zum Beispiel können Sie hier das Modul *pam_mkhomedir* einbinden. Wenn sich jetzt ein Benutzer am System anmeldet, wird für ihn automatisch ein Heimatverzeichnis angelegt, falls er keines besitzt.

5.6.2 Die PAM-Kontrollflags

Alle Module für einen bestimmten Dienst (wie zum Beispiel *sudo*) werden in der Reihenfolge abgearbeitet, in der sie in der Konfigurationsdatei des Dienstes eingetragen sind.

Über die Kontrollflags können Sie festlegen, was passieren soll, wenn ein Modul erfolgreich abgearbeitet wurde oder wenn es zu einem Fehler gekommen ist. Hier gibt es vier verschiedene Möglichkeiten, wie reagiert werden kann:

► **required**

Wenn Sie ein Modul mit dem Flag *required* einbinden, muss das Modul erfolgreich abgearbeitet worden sein, bevor PAM die Kontrolle an das aufrufende Programm zurückgibt.



Ein falsch gesetztes Flag »required« kann jegliche Anmeldung verweigern

Wenn Sie dem LDAP-Modul dieses Flag zuweisen und der LDAP-Server nicht erreichbar ist, kann es Ihnen passieren, dass der Login-Prozess für immer auf PAM wartet. Deshalb ist es immer wichtig, beim Testen von Änderungen eine aktive *root-Sitzung* zu haben.

► **requisite**

Dieses Modul muss auch immer abgearbeitet werden, aber im Gegensatz zu *required* wird die Kontrolle im Fehlerfall an die aufrufende Anwendung zurückgegeben.

► **sufficient**

Wenn ein Modul mit dem Flag *sufficient* erfolgreich abgearbeitet wurde, wird kein weiteres Modul aus der Liste der Modultypen abgearbeitet. Gibt ein Modul einen Fehler zurück, wird die Anfrage an das nächste Modul in der Liste übergeben. Das ist dann sinnvoll, wenn Sie in Ihrem System mehrere Authentifizierungsquellen eingebunden haben. Ein Beispiel dafür wäre, wenn Sie LDAP zur Authentifizierung nutzen, aber auch lokale Benutzer auf dem System haben. Wenn sich ein lokaler Benutzer anmeldet, Sie das LDAP-Modul mit dem Flag *sufficient* versehen haben und dieses Modul an erster Stelle in der Modulliste steht, würde das LDAP-Modul bei einem Fehler den Anmeldeprozess an das Modul für die lokale Anmeldung übergeben.

► **optional**

Bei diesem Flag wird ein Modul immer dann abgearbeitet, wenn vorher kein Modultyp mit *requisite* gescheitert ist und keiner mit *sufficient* erfolgreich war.

5.6.3 Argumente zu den Modulen

Beim Aufrufen der Module können Sie verschiedene Argumente an das Modul übergeben. Diese Argumente beeinflussen die Arbeitsweise des Moduls, bei dem sie eingetragen sind. Wenn Sie mehrere Authentifizierungsquellen in Ihr System eingebunden haben, wie zum Beispiel LDAP und lokale Dateien, soll ja ein Benutzer, wenn er in beiden Quellen eingetragen ist, nicht zweimal nach einem Passwort gefragt werden. In diesem Fall können Sie das Argument `use_first_pass` verwenden. Wenn bei einer Quelle die Authentifizierung für den Benutzer erfolgreich durchgeführt wurde, werden dieselben Anmeldedaten an die zweite Quelle weitergegeben, ohne dass der Benutzer weitere Eingaben tätigen muss.

5.6.4 Modulpfade

Wenn Sie ein eigenes Modul oder das Modul für eine bestimmte Hardware in Ihr System integrieren wollen, müssen Sie das Modul an einer bestimmten Stelle im Dateisystem ablegen, sodass PAM in der Lage ist, das Modul zu finden. Denn in den Konfigurationsdateien

von PAM wird immer nur der Modulname ohne Pfad eingetragen. Hier sind sich mal alle Distributionen einig: Bei allen liegen die PAM-Module im Verzeichnis */lib/security*.

5.6.5 Module und ihre Aufgaben

In diesem Abschnitt sollen einige Module etwas näher erläutert werden. Aufgrund der Vielzahl können nicht alle Module erklärt werden, aber Sie bekommen auf jeden Fall einen Einblick in die wichtigsten.

pam_unix

Hierbei handelt es sich um das Standardmodul zur Benutzerauthentifizierung. Für die Authentifizierung greift das Modul auf die Dateien */etc/passwd* und */etc/shadow* zurück. Wichtige Optionen zu dem Modul sind:

- ▶ **use_first_pass**
Bei dieser Option wird auf die Benutzereingaben eines vorherigen Moduls zugegriffen, sodass der Benutzer sein Passwort bei mehreren Authentifizierungsquellen nicht mehrfach eingeben muss.
- ▶ **use_authok**
Dieses Modul wird immer dann aktiv, wenn der Benutzer sein Passwort ändert und das geänderte Passwort durch weitere Module geleitet werden soll, wie zum Beispiel *pam_ldap* oder *pam_cracklib*.
- ▶ **nullok**
Normalerweise lässt das Modul *pam_unix* keinen Benutzer ohne Passwort zu. Durch diese Option können sich aber auch Benutzer ohne Passwort am System anmelden.

pam_cracklib

Dieses Modul überprüft neue Passwörter daraufhin, ob sie bestimmten Regeln entsprechen. Durch die Verwendung dieses Moduls können Sie verhindern, dass Benutzer unsichere Passwörter verwenden. Der Benutzer *root* ist davon nicht betroffen. Er kann weiterhin jedes beliebige Passwort an einen Benutzer vergeben.

Wichtige Optionen für dieses Modul sind:

- ▶ **difok=N**
Dieser numerische Wert legt fest, an wie vielen Stellen sich das neue vom alten Passwort unterscheiden muss, damit das neue Passwort gültig ist. Ein neues Passwort ist auf jeden Fall immer gültig, wenn sich mindestens 50 % der Zeichen von denen des alten Passworts unterscheiden.
- ▶ **minlen=N**
Über diese Option können Sie die Mindestlänge eines Passworts festlegen.

► **maxrepeat=N**

Mit diesem Wert können Sie festlegen, wie oft dasselbe Zeichen hintereinander in einem Passwort verwendet werden kann. Diese Option ist normalerweise deaktiviert, sodass beliebig viele gleiche Zeichen hintereinander verwendet werden können.

► **reject_username**

Diese Option verhindert, dass der Benutzer seinen Anmeldenamen als Passwort verwendet. Sie prüft das Passwort sowohl vorwärts als auch rückwärts.

pam_mkhome

Dieses Modul prüft bei der Anmeldung eines Benutzers, ob dieser bereits ein Heimatverzeichnis auf dem System besitzt. Hat der Benutzer noch kein Heimatverzeichnis, wird ein neues Verzeichnis für ihn angelegt. Das Modul kennt die folgenden Optionen:

► **silent**

Der Benutzer erhält keine Meldungen angezeigt.

► **umask=mask**

Hier können Sie die *umask* setzen, mit der das Heimatverzeichnis des Benutzers erstellt wird.

► **skel=/pfad/zum/skel/Verzeichnis**

Beim Erstellen des Heimatverzeichnisses können Sie gleich dafür sorgen, dass bestimmte Dateien und Verzeichnisse direkt in das neue Heimatverzeichnis des Benutzers kopiert werden. Diese Option gibt den Pfad an, aus dem die Daten kopiert werden sollen.

5.6.6 Die neuere Syntax bei der PAM-Konfiguration

Bei der neuen Schreibweise der PAM-Regeln haben Sie als Systemadministrator bessere Kontrollmöglichkeiten darüber, wie ein Benutzer authentifiziert wird. Die Regeln werden hier durch mehrere *value=action*-Parameter festgelegt. Die Liste aller Parameter zu einem Modul haben wir dabei in eckigen Klammern geschrieben. Beispiele dazu sehen Sie in Listing 5.35 in Abschnitt 5.7, »Konfiguration von PAM«.



Standard bei allen aktuellen Distributionen

Bei allen in dieser Auflage beschriebenen Distributionen wird nur noch die neue Schreibweise der PAM-Einträge verwendet. Die alte Schreibweise funktioniert aber auch noch. Wir empfehlen jedoch immer die neue Schreibweise, da Sie damit sehr viel flexibler sind.

Als Werte für *value* können Sie folgende Rückgabewerte eines Moduls verwenden:

success, *open_err*, *symbol_err*, *service_err*, *system_err*, *buf_err*, *perm_denied*, *auth_err*, *cred_insufficient*, *authinfo_unavail*, *user_unknown*, *new_authtok_reqd*, *acct_expired*, *sessi-*

on_err cred_unavail, cred_expired, cred_err, no_module_data, conv_err, authtok_err, authtok_expired, authtok_lock_busy, authtok_disable_aging, try_again, ignore, abort, authtok_recover_err, module_unknown, bad_item, maxtries, default

Der Rückgabewert `default` kann dann verwendet werden, wenn ein Rückgabewert eines Moduls nicht genau definiert ist.

`value` kann einen der folgenden Werte haben:

- ▶ `int-wert`
Durch die Angabe eines positiven `int-wert` können Sie nach einer bestimmten Aktion eine Anzahl von Modulen überspringen. Wenn Sie zum Beispiel die Authentifizierung von Benutzern über die lokale Datei `/etc/passwd`, über LDAP oder über Kerberos zulassen wollen, können Sie zum Beispiel mit `[success=2]` bei der lokalen Anmeldung die beiden Module für LDAP und Kerberos überspringen.
- ▶ `ignore`
Der Rückgabewert des Moduls wird ignoriert und nicht an die Anwendung gegeben, die eine Authentifizierung angefordert hat.
- ▶ `bad`
Damit wird der Rückgabewert des Moduls als Fehler interpretiert. Wenn das erste Modul in der Kette von Modulen diesen Wert zurückgibt, wird der Status für alle weiteren Module verwendet.
- ▶ `die`
Der Wert `die` führt zum selben Ergebnis wie `bad`, nur wird hier der PAM-Stack sofort nach dem Modul abgebrochen. Die anderen Module werden nicht mehr abgearbeitet.
- ▶ `ok`
Wenn Sie `ok` als Wert verwenden, nimmt PAM an, dass der Rückgabewert aller folgenden Module ein `ok` ist. Wenn also ein Modul ein `success` liefert, werden alle nachfolgenden Module ebenfalls ein `success` liefern. Wenn aber ein vorheriges Modul einen Fehler zurückgibt, wird der Fehler nicht überschrieben.
- ▶ `done`
Der Wert `done` ist gleichzusetzen mit `ok`, nur wird hier der PAM-Stack nicht weiter abgearbeitet, wenn ein Modul aus dem Stack diesen Rückgabewert liefert.
- ▶ `reset`
Mit dem Wert `reset` wird der gesamte Status des Stacks zurückgesetzt, und die Ermittlung des Status beginnt mit dem nächsten Modul von vorn.

Natürlich können Sie mit der neuen Syntax die alte Syntax komplett ersetzen. Damit Sie sehen, wie die alten Begriffe `required`, `requisite`, `sufficient` und `optional` auf die neue Syntax umgestellt werden, sehen Sie hier eine Gegenüberstellung der beiden Schreibweisen:

- ▶ required ist gleich [success=ok new_authtok_reqd=ok ignore=ignore default=bad].
- ▶ requisite ist gleich [success=ok new_authtok_reqd=ok ignore=ignore default=die].
- ▶ sufficient ist gleich [success=done new_authtok_reqd=done default=ignore].
- ▶ optional ist gleich [success=ok new_authtok_reqd=ok default=ignore].

5.7 Konfiguration von PAM

In diesem Abschnitt werden wir etwas näher auf die Konfiguration von PAM eingehen. In Kapitel 17, »LDAP«, wird der LDAP-Client über PAM konfiguriert. Jetzt werfen wir einen genaueren Blick auf die Konfiguration.

Bei allen Distributionen finden Sie die Konfigurationsdateien immer im Verzeichnis */etc/pam.d*. In diesem Verzeichnis finden Sie einzelne Dateien für die einzelnen Dienste. Mittlerweile nutzen alle Distributionen die Möglichkeit, durch *Include-Einträge* die Konfiguration von PAM zu vereinfachen.

In allen Distributionen gibt es zu jedem PAM-Typ eine eigene *common*-Datei, in der alle Module konfiguriert werden, die für sämtliche Anwendungen benötigt werden. Dadurch ist es möglich, durch die Änderung von nur vier Dateien alle Programme zu konfigurieren. Wenn Sie für einzelne Programme zusätzliche Module benötigen, können Sie diese in die entsprechenden Dateien optional eintragen. Um die Konfiguration an einem Beispiel zu verdeutlichen, sehen Sie in Listing 5.35 bis Listing 5.37 die Einträge für einen LDAP-Client unter Ubuntu, openSUSE und Debian, und zwar jeweils die Datei *common-auth*:

```
auth    [success=3 default=ignore]      pam_unix.so nullok_secure
auth    [success=2 default=ignore]      pam_winbind.so krb5_auth\
        krb5_ccache_type=FILE cached_login try_first_pass
auth    [success=1 default=ignore]      pam_ldap.so use_first_pass
auth    requisite                       pam_deny.so
```

Listing 5.35 Beispiel »common-auth« für Ubuntu

Hier sehen Sie, dass bei Ubuntu gleich mehrere Authentifizierungsmethoden eingebunden sind. Die Informationen können hier sowohl aus der lokalen *passwd* kommen als auch aus LDAP oder einem Windows-ADS. Am Ende der Liste steht das Modul *pam_deny*.

Dieses Modul sorgt dafür, dass für den Fall, dass keine der anderen Methoden den Benutzer authentifizieren konnte, der Zugriff auf jeden Fall verhindert wird.

Hier sehen Sie auch die neuere Schreibweise der Kontrollflags: In den eckigen Klammern können unterschiedliche Flags angegeben werden. Je nachdem, welche der Authentifizierungsquellen den Benutzer authentifiziert hat, wird ein unterschiedlicher Returncode [success=3 default=ignore] zurückgegeben.

Zum Vergleich sehen Sie in Listing 5.36 die Datei *common-auth* für openSUSE:

```
auth    required    pam_env.so
auth    sufficient  pam_unix2.so
auth    required    pam_ldap.so    use_first_pass
```

Listing 5.36 Beispiel »common-auth« für openSUSE

Wie Sie sehen, wird hier die ältere Schreibweise der Kontrollflags verwendet. Aber auch hier findet die Option `use_first_pass` Verwendung, um die Eingaben des Benutzers an die nächste Authentifizierungsmethode zu übergeben.

Jetzt fehlt uns nur noch die Datei *common-auth* von einem Debian-System mit einem konfigurierten LDAP-Client. Diese Konfiguration sehen Sie in Listing 5.37:

```
auth sufficient pam_unix.so
auth sufficient pam_ldap.so use_first_pass
auth required pam_deny.so
```

Listing 5.37 Beispiel »common-auth« für Debian

Da hier bei den Modulen `pam_unix` und `pam_ldap` jeweils das Kontrollflag `sufficient` verwendet wird, folgt im Anschluss noch das Modul `pam_deny`, um den Authentifizierungsprozess auf jeden Fall definiert abzuschließen. Eine sehr ausführliche Beschreibung aller Module und ihrer Optionen finden Sie unter folgender URL:

www.kernel.org/pub/linux/libs/pam/Linux-PAM-html

5.8 ulimit

Neben den Rechten auf Dateien und Verzeichnissen gibt es auch noch die Möglichkeit, Rechte – oder besser gesagt Beschränkungen – auf Systemressourcen zu setzen. Zum Setzen der Beschränkungen steht Ihnen das Kommando `ulimit` zur Verfügung. Bevor Sie bestimmte Beschränkungen für Benutzer oder Gruppen einrichten, sollten Sie auf jeden Fall die derzeitigen Einstellungen prüfen. Das geht mit dem Kommando `ulimit -a`. Die angezeigten Werte gelten immer für den Benutzer, mit dem Sie gerade angemeldet sind.

Listing 5.38 zeigt die Ausgaben des Kommandos `ulimit -a` für den Benutzer »root«. Die Werte, die Sie dort sehen, unterscheiden sich zwischen den einzelnen Distributionen, die Parameter sind aber überall gleich. Das Beispiel zeigt die Parameter bei openSUSE:

```
adminbuch:/etc/security # ulimit -a
core file size          (blocks, -c) 1
data seg size          (kbytes, -d) unlimited
scheduling priority     (-e) 0
file size              (blocks, -f) unlimited
```

```
pending signals          (-i) 7919
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) 869464
open files              (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority      (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes      (-u) 7919
virtual memory          (kbytes, -v) 1229520
file locks              (-x) unlimited
```

Listing 5.38 »ulimit«-Werte für »root«

Alle Parameter können an dieser Stelle nicht erklärt werden. Wenn Sie eine Übersicht über sämtliche Parameter haben wollen, finden Sie diese in der Manpage zur *bash*.

An dieser Stelle werden nur die Parameter beschrieben, die für Sie als Systemadministrator im Hinblick auf die Ressourcenbeschränkung bei Benutzern relevant sind:

- ▶ `file size`
Mit `file size` können Sie die maximale Dateigröße angeben. Dies wäre zum Beispiel eine Möglichkeit, um auf einem FTP-Server die Maximalgröße einer Datei zu beschränken, wenn anonyme Benutzer Schreibzugriff haben müssen.
- ▶ `open files`
Diesen Wert müssen Sie eventuell anpassen, wenn Sie sehr große Datenbanken auf einem Server laufen haben und dadurch sehr viele Dateien geöffnet werden müssen.
- ▶ `max user processes`
Diesen Wert sollten Sie für »normale« Benutzer niemals auf den Wert `unlimited` setzen, denn dann kann ein Benutzer so viele Prozesse starten, wie er will, und damit das System so überlasten, dass es nicht mehr reagiert.

5.8.1 Setzen der `ulimit`-Werte

Alle Parameter können Sie individuell für jeden Benutzer oder auch für eine Gruppe setzen. Die Einstellungen dafür nehmen Sie in der Datei `/etc/security/limits.conf` vor.



In dieser Datei sind bei allen Distributionen immer nur Kommentare zu finden. Mithilfe der Kommentare lassen sich die Einstellungen sehr einfach realisieren.

Pro Zeile wird in der Datei eine Beschränkung eingetragen. Sie können hier festlegen, wer bei welchem Parameter welchen Wert erhalten soll. In Listing 5.39 sehen Sie einige Beispiele:

```
ftp      soft  fsize  1048576
ftp      hard  fsize  1500000
@benutzer hard  nproc  10000
mysql    hard  nofile 2096
```

Listing 5.39 Einstellungen in der »limits.conf«

- ▶ ftp soft fsize 1048576
Für den Benutzer *ftp* wird die maximale Dateigröße auf 1 MB beschränkt. Jeder Benutzer, für den die Beschränkung gilt, kann den Wert des Softlimits bis auf den Wert des Hardlimits erhöhen. Das Hardlimit kann nur »root« verändern.
- ▶ @benutzer hard nproc 10000
Alle Mitglieder der Gruppe *benutzer* können maximal 10.000 Prozesse gleichzeitig starten. Hier wurde kein Softlimit gesetzt, also gibt es auch keine Möglichkeit für den Benutzer, den Wert zu ändern.
- ▶ mysql hard nofile 2096
Für den Benutzer *mysql* wird die Anzahl der gleichzeitig geöffneten Dateien auf 2.096 angehoben.

Bei den Hardlimits handelt es sich um Kernelparameter. Wenn Sie hier eine Änderung vornehmen, wird diese Änderung erst beim Neustart des Systems wirksam.



5.9 Abschlussbemerkung

In diesem Kapitel haben Sie verschiedene Möglichkeiten kennengelernt, wie Sie in Ihrem System Berechtigungen und Beschränkungen konfigurieren können. Aufgrund der Vielzahl an Möglichkeiten sollten Sie immer alle Optionen auf den Prüfstand stellen und diejenigen auswählen, die für Ihre Umgebung sinnvoll sind. Eine gute Planung vor der Einrichtung hilft Ihnen auch hier, später nicht alles neu konfigurieren zu müssen.

TEIL II
Aufgaben

Kapitel 6

Paketmanagement

Der Umgang mit Software – Quellcode, Paketen und Paketmanagementlösungen – steht im Vordergrund dieses Kapitels. Erfahren Sie, wie Sie Ihre selbst entwickelte Software in Pakete schnüren, Update-Pakete erstellen und das Paketmanagement Ihrer Distribution richtig einsetzen.

Zu den Aufgaben einer Paketverwaltung (engl. *packet management*) gehören das Installieren, das Aktualisieren und das Deinstallieren von Software. Schon in den Anfangsjahren von Linux wurde solch ein System erforderlich. Durch die wachsende Anzahl von Programmen der *GNU-Sammlung* konnten Linux-Systeme fast alle Aufgaben bewältigen. Lediglich das Installieren der Software sorgte für Unmut, da das Übersetzen des Quellcodes (*Kompilieren*) auf dem eigenen System oft beschwerlich war. Vorausgesetzte Software (*dependencies*¹) musste von Hand nachinstalliert werden, Updates sorgten für Konflikte mit anderer installierter Software, und die Deinstallation musste ebenfalls von Hand durchgeführt werden.

6.1 Paketverwaltung

Hier setzt die Paketverwaltung an. Sie löst Abhängigkeiten auf und versucht, notwendige Software selbstständig nachzuinstallieren. Sie entfernt Programme sauber aus dem System und prüft bei Updates, ob Überschneidungen existieren und wie diese gelöst werden können. Dafür musste die Software aber in eine entsprechende Form gebracht werden: in Pakete.

Eine Paketverwaltung besteht aus zwei Teilen. Der erste Teil ist für das Laden der Programme aus einem *Repository*² sowie für das Auflösen von Abhängigkeiten und Konflikten verantwortlich. Der zweite Teil, zum Beispiel *dpkg* (*Debian Packet Management*) oder *rpm* (*RPM Packet Management*), sorgt für die eigentliche Installation.

In diesem Kapitel erfahren Sie alles zu den zwei größten Paketverwaltungen *dpkg* und *rpm*. Sie erfahren, wie Pakete für diese Paketverwaltungen konvertiert werden, welche weiteren Programme es gibt und wie Sie selbst Pakete aktualisieren, erstellen und patchen können.

1 *Dependencies*, engl. für *Abhängigkeiten*. Damit meint man in der Paketverwaltung Software, die für den Betrieb anderer Software benötigt wird.

2 *Repository*, engl. für *Lager, Depot*.

6.1.1 rpm oder deb?

Trotz aller Vorteile, die eine Paketverwaltung bietet, besitzen Paketverwaltungen auch Nachteile. Zum einen verfügen sie zum Großteil lediglich über Binärpakete. Diese Pakete sind für eine Architektur übersetzt und für diese optimiert und auch nur auf dieser lauffähig. Dies führt zu statischen Paketen, die nicht voll an das System, auf dem sie laufen, angepasst sind. Zum anderen kommt es bei Paketverwaltungen auch zu Konflikten.



Wenn in den Paketen *Alpha* und *Beta* teilweise die gleichen Dateien enthalten sind, können Sie nicht beide Pakete gleichzeitig installieren. Falls die Aktualisierung des Pakets *Gamma* auch die Aktualisierung des Pakets *Delta* fordert, das Paket *Epsilon* aber die »alte Version« des Pakets *Delta* benötigt, ist eine Aktualisierung von *Gamma* nicht möglich. Eine gute Paketverwaltung zeichnet sich dadurch aus, dass sie diese Konflikte effektiv löst.

In der heutigen Zeit haben sich drei Konzepte durchgesetzt:

- ▶ die *rpm*-Paketverwaltung
- ▶ die *dpkg*-Paketverwaltung
- ▶ quellenbasierte Distributionen

Die ersten beiden Paketverwaltungen haben die Linux-Welt quasi in zwei Lager gespalten. Die letzte Variante findet eher selten Anwendung, aber Distributionen wie zum Beispiel *Gentoo Linux* verzichten ganz auf Binärpakete und stellen lediglich den Quellcode zur Verfügung, sodass jede Installation voll an das System angepasst wird.

Jede Distribution verfügt über ihre eigene Implementierung einer Paketverwaltung. Das hat zur Folge, dass zum Beispiel zwei auf *rpm* basierende Distributionen nicht zwingend miteinander kompatibel sein müssen.

rpm

Die Paketverwaltung *rpm* wurde ursprünglich als *Red Hat Packet Management* bezeichnet, da sie von Red Hat entwickelt wurde. Da *rpm* aber ein Teil der *Linux Standard Base* werden wollte, wurde das Projekt einfach in ein rekursives Akronym umbenannt: *RPM Packet Management*. Das Ziel vom *rpm* bestand darin, Softwarepakete sowohl für Entwickler als auch für den Anwender einfacher zu gestalten. Abhängigkeiten sollten berücksichtigt und automatisch aufgelöst werden, Redundanzen sollten vermieden werden, und das saubere Entfernen von Software sollte möglich gemacht werden. Ebenso sollte das Einspielen von Updates einfacher gestaltet werden, ebenso wie das sichere Verwalten von Konfigurationen.

1999 kam es zur Zweiteilung von *rpm*, da der Hauptentwickler Jeff Johnson Red Hat verließ und einen Fork des Projekts startete. Jetzt gibt es zwei *rpm*-Versionen: zum einen das in Suse, Red Hat, Fedora und vielen anderen Distributionen verwendete *rpm* von *rpm.org*, zum anderen den Fork von Johnson, *rpm5*, der zum Beispiel in *Alt Linux*, *ArkLinux* und *Unity Linux* Anwendung findet.

Bei *rpm*-Paketen handelt es sich um komprimierte Dateien, denen ein binärer Header vorangestellt ist. Dieser Header ist nicht komprimiert, sodass er leicht durchsucht werden kann.

Die zentrale Datei eines Pakets im *rpm*-Format ist die *SPEC*-Datei. Sie enthält alle Metainformationen des Pakets (also Informationen darüber, was in dem Paket enthalten ist, wohin es installiert wird, wie es installiert wird) sowie Informationen zum Paket selbst. Diese Datei ist in folgende acht Sektionen unterteilt:

- ▶ **Präambel**
Informationen über das Paket, die mittels *rpm* dem Benutzer angezeigt werden
- ▶ **Prep**
ein Skript, das den Quellcode-Dateibaum erzeugt
- ▶ **Build**
ein Kompilierungsskript
- ▶ **Install**
die Installationsanweisung für die Binaries – in den meisten Fällen `make install`
- ▶ **Files**
eine Auflistung aller Dateien, die im Paket enthalten sind
- ▶ **Install/Uninstall**
Skripte für die Installation und Deinstallation. Diese Sektion besteht wiederum aus vier Unterabschnitten: **pre**, **post**, **prerun** und **postrun**. Das sind Befehle, die vor/nach der Installation/Deinstallation ausgeführt werden.
- ▶ **Verify**
ein Skript zur Prüfung der Installation
- ▶ **Clean**
Befehle, die auf dem Entwicklerrechner ausgeführt werden sollen

Die Dateinamen der *rpm*-Pakete haben folgende Konvention:

```
<Bezeichnung>-<Versionsnummer>-<Revisionsnummer>.<Architektur>.rpm
```

Listing 6.1 Schema der »rpm«-Paketdateinamen

deb

Die Paketverwaltung *deb* wurde von Ian Murdock entwickelt. Die Bezeichnung *deb* bezieht sich hierbei auf die ersten drei Buchstaben der Linux-Distribution *Debian*. Die Zielsetzung von *deb* entspricht der von *rpm*, allerdings sind beide nicht miteinander kompatibel. Hierfür gibt es aber Konvertierungstools, die es erlauben, Pakete für die jeweils andere Paketverwaltung nutzbar zu machen. Ein *deb*-Paket setzt sich aus folgenden Abschnitten zusammen:

- ▶ **debian-binary**
Textdatei mit Angabe der Versionsnummer des verwendeten Paketformats

► **control**

enthält alle relevanten Informationen des Pakets, die in eigenen Dateien verwaltet werden. Die folgenden Dateien sind immer vorhanden:

- **control**
Kurzbeschreibung des Pakets und Auflistung der Abhängigkeiten
- **md5sums**
Prüfsummen der enthaltenen Dateien
- **conffiles**
Auflistung der Konfigurationsdateien im Paket
- **preinst, postinst, prerm, postrm**
Skripte, die vor oder nach der Installation/Deinstallation ausgeführt werden
- **config**
ein *debconf*-Skript, das Metainformationen für die *debconf*-Datenbank enthält
- **shlibs**
Auflistung der Programmbibliotheken

► **data**

Archiv der enthaltenen Programmdateien

Auch hier gibt es eine zentrale Datei, die für das Paket verantwortlich ist: *control*. In ihr sind sowohl alle Metainformationen zum Paket enthalten als auch die Installationsanweisungen. Dateinamen von *deb*-Paketen folgen einer definierten Syntax:

```
<Bezeichnung>_<Versionsnummer>-<Revisionsnummer>_<Architektur>.deb
```

Listing 6.2 Schema der »deb«-Paketdateinamen

6.1.2 dnf, yast, zypper oder apt?

Eine Paketverwaltung besteht nicht allein aus einem Programm. Zu ihr gehört immer (mindestens) ein Tool, das sich um das Laden der Software, das Auflösen von Abhängigkeiten und die Beseitigung von Konflikten kümmert. Jede Distribution besitzt eine eigene Umsetzung. So findet *dnf* auf Red-Hat-basierten Systemen Anwendung, *yast* und *zypper* auf Suse-basierten und *apt* auf Debian-basierten. Wenn Distributionen binärkompatibel miteinander sind, können Sie auch Pakete aus der jeweils anderen Distribution installieren.



Ubuntu- und Debian-Pakete sind zum Teil binärkompatibel, sodass Debian-Pakete auch unter einigen Ubuntu-Versionen lauffähig sind. Ebenso gilt dies für Red Hat und Suse.

Die Bedienung der einzelnen Tools unterscheidet sich erheblich voneinander. Die Debian-basierte Variante *apt* ist eine Suite mit mehreren Anwendungen, die jeweils eine Anwendung für spezielle Aufgaben zur Verfügung stellt. Die Red-Hat-Variante *dnf* kommt hingegen mit nur einem Programm aus, und die Suse-Variante *yast* kann sowohl als Kommandozeilen-

tool als auch im ASCII-Menü gesteuert werden, wobei ab openSUSE 10.2 das zusätzliche Tool *zypper* parallel zu *yast* eingesetzt werden kann.

Neben den von den Distributionen ausgelieferten Programmen können auch unabhängige Programme eingesetzt werden. Dies ist aber in den meisten Fällen nicht notwendig.



Tabelle 6.1 zeigt einen Auszug der Befehle, mit denen Sie die Paketverwaltung verschiedener Distributionen über die Kommandozeile bedienen können. Tabelle 6.2 zeigt, wie Sie lokale Pakete verwalten.

Aktion	CentOS	openSUSE Leap	Debian/Ubuntu
Paket installieren	<code>dnf install</code>	<code>yast -i</code> <code>zypper</code> <code>install</code>	<code>apt install</code>
Paket deinstallieren	<code>dnf remove</code>	<code>yast --remove</code> <code>zypper remove</code>	<code>apt remove</code>
Paketinformationen abfragen	<code>dnf info</code>	<code>zypper info</code>	<code>apt show</code>
Paket zur Datei suchen	<code>dnf provides</code>	<code>zypper wp</code>	<code>apt-file search</code>
Installierten Pakete anzeigen	<code>dnf list installed</code>	–	–

Tabelle 6.1 Auszug: Kommandozeilenbefehle für Repositories

Aktion	CentOS	openSUSE Leap	Debian/Ubuntu
Paket installieren	<code>dnf install</code>	<code>rpm -i</code>	<code>dpkg -i</code>
Paket deinstallieren	<code>dnf remove</code>	<code>rpm -e</code>	<code>dpkg -r</code>
Paketinformationen abfragen	<code>dnf info</code>	<code>rpm -qi</code>	<code>dpkg -p</code>
Paket zur Datei suchen	<code>dnf provides</code>	<code>rpm -qf</code>	–
Installierten Pakete anzeigen	<code>dnf list installed</code>	<code>rpm -qa</code>	<code>dpkg -l</code>

Tabelle 6.2 Auszug: Kommandozeilenbefehle für lokale Pakete

6.1.3 Außerirdische an Bord – alien

Es gibt Software, die nur in einer Paketform angeboten wird und nicht als Quellcode zur Verfügung steht. Um diese Software auf dem eigenen System betreiben zu können, müssen Sie das Paket umwandeln. Hier eilt Ihnen *alien* zu Hilfe. Das ursprünglich vom Debian-Ent-

wickler Christoph Lameter geschriebene Tool konvertiert Pakete in die Formate *rpm*, *deb*, *slp* (Stampede), *tgz* (Slackware) und *pkg* (Solaris). Die Installation von *alien* setzt viele weitere Pakete voraus, unter anderem auch die Kompilierungsprogramme (*gcc*, *make* etc.). Auf Systemen, die auf Debian basieren, können Sie *alien* aus den Paketquellen installieren. Leider ist *alien* weder in CentOS noch in openSUSE enthalten. Laden Sie den aktuellen Quellcode unter <http://packages.debian.org/stable/source/alien> herunter, und führen Sie nach dem Entpacken, wie in der enthaltenen Datei *INSTALL* beschrieben, die folgenden Befehle aus:

```
daniel@example:/usr/local/alien# perl Makefiles.pl
[...]
root@example:/usr/local/alien# make
root@example:/usr/local/alien# make install
```

Listing 6.3 »alien« aus dem Quellcode installieren

Oder erzeugen Sie auf einem Debian-basierten System einfach ein *rpm*-Paket mit *alien*. Laden Sie dafür die entsprechende *deb*-Datei mit `apt download alien` herunter. Wenden Sie *alien* nun mit dem Schalter `-r` (oder `--to-rpm`) an:

```
root@example:/opt# alien -r alien_8.95.5_all.deb
alien-8.95.5-2.noarch.rpm generated
```

Listing 6.4 Umwandeln von »deb« in »rpm«

alien erzeugt nun die Datei *alien-8.95.5-2.noarch.rpm*. Wie Sie dem neuen Dateinamen entnehmen können, wurde auch die Versionsnummer angepasst. Falls Sie die eigentliche Versionsnummer des Pakets behalten wollen, verwenden Sie den Parameter `-k` beim Aufruf von *alien*. Falls Sie ein Paket auf dem System konvertieren, auf dem die Software installiert werden soll, können Sie diese mit dem Schalter `-i` auch direkt von *alien* installieren lassen.



Speicherort installierter Pakete

Installierte Pakete auf Systemen mit *deb*-Paketverwaltung finden Sie im Verzeichnis `/var/cache/apt/archives/`. Bei openSUSE Leap finden Sie die installierten Pakete unter `/var/cache/zypp/packages/<Repository>:<RepoName>/rpm/` und bei CentOS unterhalb von `/var/cache/dnf`.

6.2 Pakete im Eigenbau

Neben der Vielzahl an Software, die über die Repositories bezogen werden kann, gibt es auch Software, die lediglich als Quellcode zur Verfügung steht. Um solche Programme auf Ihrem System betreiben zu können, müssen Sie sie kompilieren. Hierfür verwenden Sie den Ihnen vermutlich bereits bekannten Dreisprung:

```
daniel@example:/usr/local/packet# ./configure
daniel@example:/usr/local/packet# make
daniel@example:/usr/local/packet# sudo make install
```

Listing 6.5 Kompilierung im Dreisprung

Dies ist natürlich eine adäquate Möglichkeit, Software auf »einem« System zu verwenden. Wenn Sie aber mehrere Systeme der gleichen Architektur mit dem gleichen Betriebssystem verwenden, können Sie den Quellcode auch in ein Paket schnüren, um so die Software einfach zu verteilen, eine Update-Sicherheit zu erzeugen oder auch Patches effektiv zu verteilen. In diesem Abschnitt erfahren Sie alles zu Paketen: wie Sie diese aus einem *tarball*³ erzeugen, wie Sie Patches einspielen und wie Sie Update-Pakete erstellen.

6

6.2.1 Vorbereitungen

Damit Sie Pakete erzeugen können, müssen einige Pakete auf Ihrem System vorhanden sein, die zur Erstellung notwendig sind: die sogenannte *Build-Umgebung*. Keine Sorge, Sie müssen die benötigten Pakete nicht einzeln identifizieren und separat installieren. In den Paketquellen sind Metapakete oder Gruppen vorhanden, die alle benötigten Pakete enthalten. Die Befehle auf Listing 6.6 installieren die benötigten Pakete:

```
# Debian/Ubuntu
root@ubuntu:~$ apt install build-essential

# CentOS
[root@centos ~]# dnf -y groupinstall Development\ Tools

# openSUSE Leap
leap:~> sudo zypper install -t pattern devel_C_C++ devel_basis devel_rpm_build
```

Listing 6.6 Build-Umgebung installieren

6.2.2 Am Anfang war das Makefile

Software, die nicht in den Repositories zur Verfügung steht, wird in der Regel von den Entwicklern als *tgz*- oder auch *tar.gz*-Datei zur Verfügung gestellt. Aus dieser Datei entpacken Sie den Quellcode und kompilieren dann das Programm. Um selbst geschriebene Software so zur Verfügung zu stellen, muss diese erst in die richtige Form gebracht werden. Dazu dient das *Makefile*, auf das das *configure*-Skript zugreift, das das Programm entsprechend Ihrem System verarbeitet. Im Grunde genommen sind Makefiles nichts weiter als Textdateien, in denen der Übersetzungsprozess von Programmen formalisiert enthalten ist.

³ *tarball* (*tape archive ball*) – Archivformat, in dem oft Quellcode ausgeliefert wird.

Darüber hinaus stellt das Makefile eine Intelligenz zur Verfügung, die bereits vorhandene und kompilierte Abhängigkeiten (oder bei Updates auch eigenen Code) erkennt und diese nicht erneut kompiliert. Das Erstellen eines solchen Makefiles von Hand kann bei größeren Projekten mit vielen Abhängigkeiten schnell in unüberschaubare Arbeit ausarten. Um effizient Makefiles zu erstellen, gibt es die *GNU autotools*, die Ihnen die Arbeit zu einem Großteil abnehmen. Folgende Arbeitsschritte müssen Sie durchlaufen, um ein eigenes *tgz* zu erstellen:

1. Quellcode erstellen
2. *Makefile.am* erstellen
3. *autoscan* – *configure.scan* in *configure.in* umwandeln
4. *aclocal* – Anpassung für die Sprachumgebung
5. *autoconf* – Verarbeitung von Konfigurationsdateien
6. *autoheader* – Verarbeitung von zusätzlichem Quellcode
7. *automake* – Erzeugen des Makefiles

In diesem Abschnitt zeigen wir Ihnen, wie Sie das allseits beliebte Programm *helloworld* von seinem C-Code-Dasein befreien und als *tgz*-Datei zur Verfügung stellen. Wenn noch nicht vorhanden, installieren Sie das Paket *autoconf* aus den Paketquellen (in der Regel ist das Paket aber Bestandteil der Build-Umgebung). Erstellen Sie zunächst ein Verzeichnis *hello-1.0* in Ihrem Homeverzeichnis. In diesem Verzeichnis erstellen Sie das *helloworld.c*-Programm mit folgendem Quellcode:

```
/* Simple "Hello World!" program in C */
#include <stdio.h>
int main()
{
    printf("Hello world! Once again... \n");
}
```

Listing 6.7 »helloworld.c«



Erstellung als Benutzer

Die Erstellung von Paketen (*tgz*, *rpm* oder *deb*) sollte immer als normaler Benutzer ausgeführt werden. Da *root* Vollzugriff auf alle Dateien hat, könnten während der Installation eines von ihm erstellten Pakets Dateien überschrieben oder gelöscht werden, die auf dem System benötigt werden. Die Installation des gepackten Programms selbst muss hingegen von *root* durchgeführt werden, da Benutzern die entsprechenden Rechte fehlen.

Das soeben erstellte Programm dient als zu verpackende Basis. Erstellen Sie nun die Datei *Makefile.am* mit folgendem Inhalt:

```
bin_PROGRAMS = helloworld
helloworld_SOURCES = helloworld.c
```

Listing 6.8 »Makefile.am«

Die Datei *Makefile.am* wird von den *autotools* ausgewertet. In ihr werden der Name des Programms (`bin_PROGRAMS`) und dessen Quellcode (`helloworld_SOURCES`) deklariert. Bei größeren Projekten werden hier alle programmrelevanten Dateien aufgeführt und auch alle Bibliotheken. *autoscan* analysiert das *Makefile.am* und ihren Quellcode. Es erstellt dabei die Datei *configure.scan*, die als Basis zur Erstellung der *configure.ac* dient (siehe Listing 6.9):

```
#                                     -*- Autoconf -*-
# Process this file with autoconf to produce a configure script.
AC_PREREQ([2.71])
AC_INIT([FULL-PACKAGE-NAME], [VERSION], [BUG-REPORT-ADDRESS])
AC_CONFIG_SRCDIR([helloworld.c])
AC_CONFIG_HEADERS([config.h])

# Checks for programs.
AC_PROG_CC

# Checks for libraries.

# Checks for header files.

# Checks for typedefs, structures, and compiler characteristics.

# Checks for library functions.

AC_CONFIG_FILES([Makefile])
AC_OUTPUT
```

Listing 6.9 »configure.scan«

Benennen Sie die Datei entsprechend um, und passen Sie die Parameter wie in Listing 6.10 dem *helloworld.c*-Programm an:

```
AC_PREREQ([2.69])
AC_INIT([helloworld], [1.0], [bugs@example.com])
AC_CONFIG_SRCDIR([helloworld.c])
AM_INIT_AUTOMAKE([1.9 foreign])
AC_PROG_CC
AC_CONFIG_FILES([Makefile])
AC_OUTPUT
```

Listing 6.10 »configure.ac«

Achten Sie darauf, die Zeile `AC_CONFIG_HEADERS([config.h])` zu entfernen, da unser Programm keine Header-Datei besitzt. Da Sie weiter mit den *autotools* arbeiten, müssen Sie die vierte Zeile hinzufügen, die von *autoscan* nicht erstellt wurde. Führen Sie anschließend die Befehle `aclocal` und `autoconf` aus. Diese ergänzen und erweitern die Struktur des *tgz*. Führen Sie nun `autoheader` aus, das mit einer Fehlermeldung endet (siehe Listing 6.11):

```
daniel@example:/home/daniel/hello-1.0# autoheader
autoheader: error: AC_CONFIG_HEADERS not found in configure.ac
```

Listing 6.11 Fehlermeldung: »autoheader«

Obwohl der Befehl eine Fehlermeldung ausgibt, nimmt er wichtige Konfigurationen vor. Anschließend beenden Sie die Verarbeitung mit `automake`, das neben dem *configure*-Skript das passende *Makefile* erzeugt:

```
daniel@example:/home/daniel/hello-1.0# automake --add-missing
```

Listing 6.12 Fehlende Dateien ergänzen

Der Parameter `--add-missing` gibt an, dass *automake* fehlende Standarddateien hinzufügen soll. Jetzt stehen Ihnen alle benötigten Dateien zur Verfügung. Packen Sie diese mittels *tar* in ein Archiv, das Sie auf jedem System entpacken und kompilieren können:

```
daniel@example:/home/daniel/# tar -cavhf hello-1.0.tar.gz hello-1.0/
```

Listing 6.13 Ein »tar«-Archiv erstellen

Nach dem bekannten Dreisprung können Sie das Programm einfach über den Befehl `helloworld` aufrufen und erhalten die erwartete Ausgabe: »Hello World! Once again ...«. Beachten Sie, dass der Befehl `make install` immer als *root* ausgeführt werden muss!

6.2.3 Vom Fellknäuel zum Paket

Eine noch elegantere Variante, um das Programm *helloworld* zur Verfügung zu stellen, besteht darin, es in ein *rpm*- oder *deb*-Paket zu hüllen. In diesem Abschnitt zeigen wir Ihnen die minimalen Anforderungen, um ein Paket erzeugen zu können. Bitte beachten Sie, dass noch weitere Konfigurationen notwendig sind, wenn Sie Pakete in die Paketverwaltung Ihrer Distribution hochladen wollen. (Lesen Sie dafür die *Guidelines* der Distribution.)

deb-Pakete erzeugen: »dpkg-buildpackage«

Zum Erzeugen von *deb*-Paketen wird eine gewisse Verzeichnisstruktur vorausgesetzt. Erzeugen Sie daher die benötigten Verzeichnisse:

```
daniel@ubuntu:~$ mkdir -p build/hello-1.0/debian
```

Listing 6.14 Build-Umgebung erzeugen

Legen Sie nun die Datei *control* im Verzeichnis *debian* an. Diese enthält wichtige Informationen zum Paket. In Listing 6.15 sehen Sie die Werte, die mindestens gesetzt sein müssen:

```
Source: hello
Build-Depends: debhelper (>= 9)
Maintainer: Daniel van Soest <daniel@example.com>
Standards-Version: 3.9.5
```

```
Package: hello
Architecture: any
Description: Easy to use "helloworld" Program written in C.
    Need some sunshine? Just go and run this nifty little program.
```

Listing 6.15 Inhalt von »debian/control«

Bitte passen Sie den *Maintainer* (engl. »to maintain«, dt. »instandhalten« – Autor des distributionsspezifischen Pakets) mit gültigen Werten an (in Fettschrift dargestellt). Beachten Sie, dass die erste Zeile der Direktive *Description* die Kurzbeschreibung darstellt. Die folgenden Zeilen beginnen stets mit einem Leerzeichen – all diese Zeilen werden als ausführliche Beschreibung ausgewertet. Eine weitere Datei, die im Verzeichnis *debian* zwingend vorhanden sein muss, ist *changelog*. Erstellen Sie diese mit dem Inhalt aus Listing 6.16:

```
hello (1.0-1) unstable; urgency=low

    * Initial release

-- Daniel van Soest <daniel@example.com> Tue, 27 Dec 2022 15:50:37 +0200
```

Listing 6.16 Inhalt von »debian/changelog«

Auch hier müssen Sie in der letzten Zeile den *Maintainer* korrigieren und das Datum und die Uhrzeit anpassen. Zusätzlich werden noch weitere Dateien benötigt, die wir in einem Schwung mit den Befehlen aus Listing 6.17 erzeugen:

```
daniel@ubuntu:~/build/hello-1.0$ echo '9' > debian/compat
daniel@ubuntu:~/build/hello-1.0$ echo 'helloworld /usr/bin/' > debian/hello.install
daniel@ubuntu:~/build/hello-1.0$ echo -e '%:\n\tdh $@' > debian/rules
```

Listing 6.17 Weitere Dateien anlegen und befüllen

Zu guter Letzt müssen Sie noch die bereits kompilierte Version des Programms mit `cp /usr/local/bin/helloworld ~/build/hello-1.0` ins Stammverzeichnis der Build-Umgebung kopieren. Damit sind alle benötigten Dateien vorhanden, und Sie können das *deb*-Paket mit dem Befehl aus Listing 6.18 erzeugen:

```
daniel@ubuntu:~/build/hello-1.0$ dpkg-buildpackage -rfakeroot
[...]
```

```
daniel@ubuntu:~/build/hello-1.0$ ls -lha ../
total 40K
drwxrwxr-x 3 daniel daniel 4,0K Dez 27 14:53 .
drwxr-x--- 8 daniel daniel 4,0K Dez 27 14:53 ..
drwxrwxr-x 3 daniel daniel 4,0K Dez 27 14:51 hello-1.0
-rw-rw-r-- 1 daniel daniel 6,0K Dez 27 14:53 hello_1.0-1_amd64.buildinfo
-rw-rw-r-- 1 daniel daniel 1,6K Dez 27 14:53 hello_1.0-1_amd64.changes
-rw-r--r-- 1 daniel daniel 2,5K Dez 27 14:53 hello_1.0-1_amd64.deb
-rw-rw-r-- 1 daniel daniel 485 Dez 27 14:53 hello_1.0-1.dsc
-rw-rw-r-- 1 daniel daniel 3,7K Dez 27 14:53 hello_1.0-1.tar.gz
-rw-r--r-- 1 daniel daniel 3,0K Dez 27 14:53 hello-dbgSYM_1.0-1_amd64.ddeb
```

Listing 6.18 »deb«-Paket erzeugen mit »dpkg-buildpackage«



Ubuntu → Debian

Auf Ubuntu-Systemen erstellte Pakete lassen sich seit 2018 leider nicht mehr auf Debian-Systemen installieren. Der Vorgang bricht mit folgendem Fehler ab:

```
root@debian:~# dpkg -i hello_1.0-1_amd64.deb
dpkg-deb: Fehler: Archiv »hello_1.0-1_amd64.deb« verwendet unbekannte
Komprimierung für Element »control.tar.zst«, Abbruch
```

Leider versteht Debian keine zst-Komprimierung. Damit die Pakete dennoch installiert werden können, müssen Sie vor dem Erstellen in der Datei *debian/rules* am Anfang der Datei folgende Zeilen ergänzen:

```
override_dh_builddeb:
    dh_builddeb -- -Zgzip
```

Bitte beachten Sie, dass die Einrückung mit der Tab-Taste erfolgen muss und nicht aus Leerzeichen bestehen darf! So erzeugte Pakete lassen sich auch auf Debian-Systemen installieren.

RPM-Pakete erzeugen: »rpmbuild«

Um ein RPM-Paket aus dem Programm *helloworld* zu erzeugen, müssen Sie zunächst eine Build-Umgebung erstellen. Führen Sie dafür die Befehle aus Listing 6.19 aus:

```
[daniel@centos ~]$ mkdir -p ~/rpmbuild/{BUILD,RPMS,SOURCES,SPECS,SRPMS}
[daniel@centos ~]$ echo '%_topdir %(echo $HOME)/rpmbuild' > ~/.rpmmacros
```

Listing 6.19 Build-Umgebung erstellen: »CentOS«

Mit diesen Befehlen erzeugen Sie alle benötigten Unterverzeichnisse und erstellen eine Makro-Datei, damit das *rpmbuild* auch weiß, was zu tun ist. Kopieren Sie anschließend die gepackten Quellen in den erzeugten Ordner *SOURCES*. Für das Beispielprogramm *helloworld* genügt der Befehl aus Listing 6.20:

```
[daniel@centos rpmbuild]$ cp ../hello-1.0.tar.gz SOURCES/
```

Listing 6.20 Quellen kopieren nach »SOURCES«

Zum Erzeugen eines RPM-Pakets wird stets eine sogenannte *spec*-Datei benötigt. Diese enthält alle Informationen zum Paket und spezifiziert, wie das Paket erzeugt werden soll. Öffnen Sie nun die Datei *SPECS/hello.spec* mit einem Editor. Auf Suse-Systemen wird die Build-Umgebung erkannt und direkt die passende Template-Datei geöffnet. Auf CentOS-Systemen müssen diese manuell anlegen. In dieser Datei müssen aber immer mindestens die Zeilen angepasst werden, die in Listing 6.21 in Fettschrift dargestellt sind:

```
Name:          hello
Version:       1.0
Release:       0
Summary:       Easy to use "helloworld" Program written in C.
License:       GPLv2
Group:         Development/Tools
Source:        %name-%version.tar.gz
[...]
%description
# Easy to use "helloworld" Program written in C.
# Need some sunshine? Just go and run this nifty little program.
%prep
%setup -q

%build
%configure
%make_build

%install
%make_install

%post
%postun

%files
/usr/bin/helloworld

%license
%doc

%changelog
# Initial release
```

Listing 6.21 Minimaler Inhalt in »SPECS/hello.spec«, hier in Fettschrift

Falls in Ihrem Paket die Standarddokumentation vorhanden ist, wie zum Beispiel *README* oder *COPYING*, müssen Sie diese unter `doc` mit Leerzeichen getrennt angeben: `%doc <FILE1> <FILE2> <...>`. Ebenso sollten Sie eine ausführliche Beschreibung unter `%description` angeben – das vorangestellte Rautezeichen im Beispiel aus Listing 6.21 ist im übrigen verpflichtend und gibt an, wie viele Zeilen zu der Beschreibung gehören. Falls Sie Aktualisierungen herausbringen, sollten die Änderungen unter `%changelog` erläutert werden. Dies alles gehört zu einem guten Stil und ist ein »Muss«. Wenn Sie alle Angaben eingetragen haben, können Sie nun mit dem Programm `rpmbuild` das Paket so erzeugen, wie in Listing 6.22 dargestellt:

```
[daniel@centos rpmbuild]$ rpmbuild -ba SPECS/hello.spec
[...]
[daniel@centos rpmbuild]$ find ./ -name '*.rpm'
./RPMS/x86_64/hello-1.0-1.x86_64.rpm
./RPMS/x86_64/hello-debugsource-1.0-1.x86_64.rpm
./RPMS/x86_64/hello-debuginfo-1.0-1.x86_64.rpm
./SRPMS/hello-1.0-1.src.rpm
```

Listing 6.22 Erstellung des Pakets

Wie Sie in Listing 6.22 sehen, war die Erstellung des Pakets erfolgreich. Dieses befindet sich dann, je nach Architektur, unterhalb von `RPMS/<ARCH>/`.

6.2.4 Patchen mit `patch` und `diff`

Das klassische Patchen von Software, das Sicherheitslücken oder Bugs behebt, findet heutzutage kaum noch Anwendung. In den meisten Fällen werden Updates binnen kürzester Zeit als Paket angeboten und in die Repositorys übernommen. Dennoch gibt es Patches, die eventuelle Verbesserungen mit sich bringen oder sogar in Paketen enthalten sind. Letzteres tritt oft ein, wenn der Maintainer nicht das ganze Paket neu erstellt, sondern lediglich das Update in Form eines Patches hinzugefügt hat. Ein Patch besteht aus einer *diff*-Datei. Das Programm *diff* erstellt eine Übersicht über die Unterschiede zweier Dateien. Diese wird in einer speziellen Syntax dargestellt, sodass hinzugefügte, gelöschte oder veränderte Zeilen separat dargestellt werden. Für ein einfaches Beispiel erstellen Sie zwei Textdateien, wie sie in Tabelle 6.3 gezeigt werden.

datei01.txt	datei02.txt
Ein einfaches Beispiel, das die Funktionsweise von 'diff' darstellen soll.	Ein einfaches BEISPIEL, das die Funktionsweise des gut funktionierenden Programms 'diff' darstellen soll.

Tabelle 6.3 Ausgangsdateien

Erstellen Sie einen *diff* der Dateien über den nachstehenden Befehl:

```
daniel@example:~# diff -Naur datei01.txt datei02.txt > datei01.patch
```

Listing 6.23 »diff«-Erzeugung

Die Schalter *-Naur* weisen *diff* an, die Unterschiede der *datei01.txt* zur *datei02.txt* vollständig zu ermitteln und im C-Format darzustellen. Die Ausgabe wird in die Datei *datei01.patch* geschrieben, die folgenden Inhalt hat:

```
--- datei01.txt 2022-12-28 12:02:38.109147670 +0000
+++ datei02.txt 2022-12-28 12:03:04.140788884 +0000
@@ -1,3 +1,4 @@
-Ein einfaches Beispiel,
-das die Funktionsweise von
+Ein einfaches BEISPIEL,
+das die Funktionsweise
+des gut funktionierenden Programms
 'diff' darstellen soll.
```

Listing 6.24 »diff« der »datei01.txt« zu »datei02.txt«

In den ersten beiden Zeilen werden die Ursprungsdateien definiert. Durch die dreifachen Plus- und Minuszeichen wird angegeben, von welcher zu welcher Datei der *diff* erstellt wurde. Die dritte Zeile, die von At-Zeichen (@) umgeben ist, zeigt an, wo sich der entsprechende Block in beiden Dateien befindet und, durch ein Komma getrennt, wie lang er in der jeweiligen Datei ist. Anschließend folgen die Zeilen, die durch Symbole kategorisiert sind. Das Programm *diff* kennt drei Kategorien:

- ▶ **Hinzugefügte Zeilen** = voranstehendes Pluszeichen (+)
- ▶ **Gelöschte Zeilen** = voranstehendes Minuszeichen (-)
- ▶ **Nicht veränderte Zeilen** = ohne voranstehendes Zeichen

In einer *diff*-Datei können auch mehrere Dateien und deren Veränderungen aufgelistet sein. In ihr wird nie der gesamte Inhalt einer Datei aufgelistet, sondern lediglich die Veränderungen und deren direkte Umgebung. Daher kann die dritte Zeile des Beispiels auch mehrfach vorkommen und somit mehrere geänderte Blöcke einer Datei referenzieren.

Diese Dateien bilden die Grundlage eines Patches. Das Programm *patch* wird angewandt, um diese Veränderungen einzuspielen. Der Standardaufruf von *patch* lautet:

```
root@example:~# patch -Np0 -i datei01.patch
```

Listing 6.25 Patch einspielen

Das Programm spielt nun die Änderungen aus der Datei *datei01.patch* ein. Der Schalter *-N* gibt dabei an, dass die Richtung aus der *diff*-Datei verwendet werden soll. Der Schalter *p0*

weist *patch* an, die Verzeichnisstruktur ab dem Level »O« anzuwenden, und der Schalter *-i* verlangt die Patch-Datei. Software-Patches werden meist mit einer Verzeichnisstruktur erzeugt. Erstellen Sie folgende Verzeichnisstruktur:

```
build/  
|-- orig  
|  `-- datei.txt  
`-- orig.patch  
    `-- datei.txt
```

Listing 6.26 Patch-Verzeichnisstruktur

Kopieren Sie anschließend die *datei01.txt* nach *build/orig/datei.txt* und die Datei *datei02.txt* nach *build/orig.patch/datei.txt*, und stellen Sie deren ursprünglichen Inhalt wieder her. Erstellen Sie nun den neuen *diff* mit folgendem Befehl:

```
daniel@example:~# diff -Naur build/orig/datei.txt \  
build/orig.patch/datei.txt > datei.patch
```

Listing 6.27 Neue Patch-Datei erstellen

Spieren Sie den angelegten Patch ein:

```
daniel@example:~# patch -Np0 -i datei.patch  
patching file build/orig/datei.txt
```

Listing 6.28 Patch-Datei einspielen

Die Datei *orig/datei.txt* entspricht jetzt der Datei *orig.patch/datei.txt*. Die Besonderheit des Parameters *p0* wird im nächsten Beispiel deutlich. Hier wird der gleiche Patch erneut eingespielt, diesmal aber von einer tieferen Verzeichnisebene aus:

```
daniel@example:~/build# patch -Np1 -i ../datei.patch  
patching file orig/datei.txt
```

Listing 6.29 Patch-Datei aus tieferer Verzeichnisebene einspielen

Da sich die Verzeichnisstruktur fest in der Patch-Datei befindet, muss *patch* über den Parameter 1 angewiesen werden, den Pfad um den ersten Teil der Verzeichnisstruktur zu kürzen, damit die Dateien auch gefunden werden.



Länge der Pfade

Das Programm *patch* prüft vor jedem Durchlauf, ob bereits ein Patch eingespielt wurde. Dies kann Ihnen zum Verhängnis werden, wenn der Pfad zur »neuen« Datei nicht länger ist als der zur »alten«. Beachten Sie dies nicht, wird der erstellte *diff* nicht von *patch* eingespielt.

6.2.5 Updates sicher konfigurieren

Zu den Nachteilen einer Paketverwaltung gehört, dass sie als *root* ausgeführt wird und somit Vollzugriff auf alle Dateien hat. So kann es vorkommen, dass beim Aktualisieren von Paketen vorhandene Konfigurationsdateien überschrieben werden. Ein weiterer Nachteil besteht darin, dass Abhängigkeiten von den Paketverwaltungen nur so weit verarbeitet werden können, wie diese von den Maintainern gepflegt wurden. Diese Umstände führen teilweise zu Störungen, deren Ursache nicht leicht auffindbar ist. In diesem Abschnitt erfahren Sie, wie Sie den GAU vermeiden können.

Der »hold«-Status

Leider kommt es immer wieder vor, dass die Weiterentwicklung von Software länger dauert als geplant. Dies kann zu Konflikten führen.

Nehmen wir an, die Entwicklung der Software *Alpha* steht still. Hingegen wird die Software *Beta*, von der *Alpha* abhängt, stetig weiterentwickelt. Wenn nun »alte« Funktionen aus der Software *Beta* nicht weiter in die »neuen« Versionen übernommen werden, kann die Software *Alpha* nicht mehr ordnungsgemäß arbeiten.



Eine Paketverwaltung löst solche Probleme meist auf und unterbindet das weitere Einspielen von Updates für die entsprechende Software. Leider kommt es vor, dass Entwickler keine Versionsangaben für abhängige Software einpflegen. Dann kommt es zum GAU (dem größten anzunehmenden Unfall). Um diesen zu verhindern, haben Sie die Möglichkeit, einzelne Pakete aus den Updates herauszulösen, sodass die von Ihnen benötigte Software zwar im veralteten, aber dafür lauffähigen Zustand bleibt. Hierfür wurde in den *deb*-Paketverwaltungen der *hold*-Status und unter der *rpm*-Paketverwaltung der *lock*-Status implementiert.

Wie Sie das Paket in den verschiedenen Paketverwaltungen sperren können zeigen wir Ihnen in Listing 6.30 – Beachten Sie dabei, dass bei CentOS das vorher Pakete installiert werden müssen!

```
### Debian/Ubuntu
root@debian:~# echo "vsftpd hold" | dpkg --set-selections

### openSUSE Leap
daniel@leap:~> sudo zypper addlock vsftpd
Specified lock has been successfully added.

### CentOS
[daniel@centos ~]$ sudo dnf install dnf-plugin-versionlock
[...]
[daniel@centos ~]$ sudo dnf versionlock vsftpd
Letzte Prüfung auf abgelaufene Metadaten: vor 1:48:15 am Di 27 Dez 2022 16:26:52 CET.
Versionssperre wird hinzugefügt zu: vsftpd-0:3.0.3-48.el9.*
```

```
Versionssperre wird hinzugefügt zu: vsftpd-0:3.0.3-47.el9.*
Versionssperre wird hinzugefügt zu: vsftpd-0:3.0.3-49.el9.*
```

Listing 6.30 Pakete sperren

Übersicht der gesperrten Pakete

Eine Übersicht der gesperrten Pakete erhalten Sie über die folgenden Befehle:

```
### Debian / Ubuntu:
root@debian:~# dpkg --get-selections | grep hold
vsftpd                                hold

### openSUSE Leap
daniel@leap:~> zypper locks
# | Name   | Type   | Repository | Comment
--+-+-----+-----+-----+-----
1 | vsftpd | package | (any)      |

### CentOS
[daniel@centos ~]$ dnf versionlock
Letzte Prüfung auf abgelaufene Metadaten: vor 0:45:51 am Di 27 Dez 2022 17:31:22 CET.
vsftpd-0:3.0.3-48.el9.*
vsftpd-0:3.0.3-47.el9.*
vsftpd-0:3.0.3-49.el9.*
```

Listing 6.31 Übersicht gesperrter Pakete

Konfigurationsdateien

Prüfen Sie die aktuellen Paketverwaltungen daraufhin, ob Veränderungen an vorhandenen Konfigurationsdateien vorgenommen wurden, und legen Sie entsprechend Sicherungskopien an, sodass bei einem Update Ihre »alte« Konfiguration nicht blind überschrieben wird.

rpm

Bei der Installation untersucht *rpm*, ob es Änderungen an der Konfigurationsdatei gibt. Je nachdem, wie das Paket erstellt wurde, wird Ihre aktuelle Konfigurationsdatei ersetzt und eine Sicherheitskopie erzeugt (Suffix: *.rpmsave* oder *.rpmorig*), oder Ihre Konfigurationsdatei bleibt bestehen, und die »neue« Konfigurationsdatei wird mit dem Suffix *.rpmnew* angelegt.



».rpmsave«, »rpmorig« und »rpmnew«

Kontrollieren Sie nach »großen« Updates, ob diese Dateien existieren, und passen Sie gegebenenfalls Ihre Konfiguration an!

deb

Bei *deb*-basierten Systemen wird ein anderer Ansatz verfolgt. Hier wird ebenfalls geprüft, ob eine lokale Konfigurationsdatei existiert. Wird mit dem »neuen« Paket eine Konfigurationsdatei ausgeliefert, wird Ihnen die Wahl gelassen (siehe Abbildung 6.1).

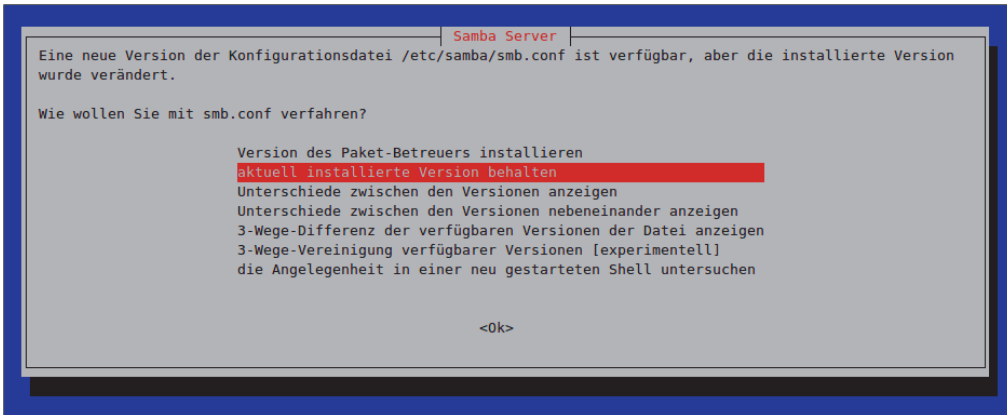


Abbildung 6.1 Auswahl zur Änderung der Konfigurationsdatei »samba«

6.3 Updates nur einmal laden: Cache

Bandbreite ist Zeit, und Zeit ist kostbar. Es ist ärgerlich, wenn ein und dasselbe Update von zehn (oder mehr) Ihrer Server Ihre Internetleitung minutenlang bis zum Anschlag füllt. Vor allem dann, wenn es doch ausreichen würde, das Update einmal aus dem Internet herunterzuladen und in Ihrem Netzwerk zu verteilen!

6.3.1 deb-basierte Distributionen: apt-cacher-ng

Der *apt-cacher-ng* stellt umfangreiche Funktionen zur Verfügung. Mit ihm können Sie einen effizienten Cache betreiben, er lässt multiple Zugriffe zu und bietet Ihnen auch noch Auswertungsmöglichkeiten.

6.3.2 Installation

Installieren Sie das Paket wie gewohnt mit `apt install apt-cacher-ng` aus den Paketquellen. Die benötigten Abhängigkeiten werden automatisch mit installiert. Auf Ubuntu-Systemen muss dafür die *universe*-Paketquelle aktiviert sein.

Bei der Installation erhalten Sie den Sicherheitshinweis aus Abbildung 6.2, den Sie genau befolgen sollten. Wir empfehlen Ihnen daher die Vorauswahl *Nein* zu bestätigen.

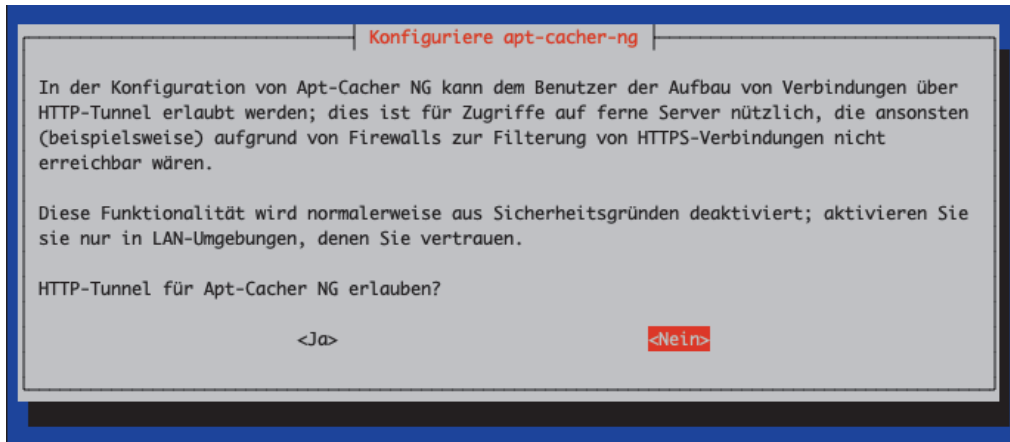


Abbildung 6.2 Hinweis bei der Installation des »apt-cacher-ng«

6.3.3 Konfiguration

Nach der Installation des Pakets *apt-cacher-ng* finden Sie unter */etc/apt-cacher-ng* die Konfigurationsdatei *acng.conf* des Dienstes. In der Standardkonfiguration sind bereits einige Parameter gesetzt.

Listing 6.32 zeigt den Inhalt der Datei ohne Kommentarzeilen:

```
CacheDir: /var/cache/apt-cacher-ng
LogDir: /var/log/apt-cacher-ng
SupportDir: /usr/lib/apt-cacher-ng
Remap-debrep: file:deb_mirror*.gz /debian ; file:backends_debian # Debian Archives
Remap-uburep: file:ubuntu_mirrors /ubuntu ; file:backends_ubuntu # Ubuntu Archives
Remap-klxrep: file:kali_mirrors /kali ; file:backends_kali # Kali Linux Archives
Remap-cygwin: file:cygwin_mirrors /cygwin
Remap-sfnet: file:sfnet_mirrors
Remap-alexrep: file:archlx_mirrors /archlinux # ; file:backend_archlx # Arch Linux
Remap-fedora: file:fedora_mirrors # Fedora Linux
Remap-epel: file:epel_mirrors # Fedora EPEL
Remap-slrep: file:sl_mirrors # Scientific Linux
Remap-gentoo: file:gentoo_mirrors.gz /gentoo ; file:backends_gentoo
Remap-secdeb: security.debian.org security.debian.org/debian-security deb.[...]
ReportPage: acng-report.html
ExThreshold: 4
FollowIndexFileRemoval: 1
LocalDirs: acng-doc /usr/share/doc/apt-cacher-ng
```

Listing 6.32 Auszug der Standardwerte: »acng.conf«

Dabei sind folgende dienstspezifische Parameter konfiguriert:

- ▶ **CacheDir**
Verzeichnis, in dem der Cache abgelegt wird
- ▶ **LogDir**
Verzeichnis, in dem die Log-Dateien abgelegt werden
- ▶ **Port**
Definition des TCP-Ports, auf dem der Dienst lauscht (Standard: 3142)
- ▶ **ExTreshold**
Angabe in Tagen, ab wann ein nicht referenziertes Paket als abgelaufen betrachtet wird

Zusätzlich können Sie auch nachstehende Parameter konfigurieren:

- ▶ **VerboseLog, Debug, ForeGround**
Parameter zur Fehleranalyse
- ▶ **Offlinemode**
steuert das Verhalten, wenn keine Internetverbindung existiert
- ▶ **StupidFs**
steuert zusätzliche Maßnahmen, wenn der Cache auf einem NTFS- oder FAT-Dateisystem betrieben wird
- ▶ **Proxy**
Darüber kann ein übergeordneter Proxy-Server angegeben werden, falls Ihr Server nicht direkt mit dem Internet kommunizieren kann.

Neben diesen dienstspezifischen Konfigurationen findet das *Remapping* über den Parameter *Remap-* statt. In jeder Zeile wird definiert, für welche Anfragen eine entsprechende Datei (Angabe nach dem Parameter *file*) geladen werden soll. In diesem Zusammenhang spricht man auch von *Backends*. Wie Sie Listing 6.32 entnehmen können, sind neben den Ubuntu-Quellen auch direkt Debian-, cygwin-, Arch-Linux-, Fedora- und weitere Quellen definiert. Zu der reinen Konfiguration des *Remappings* gehört aber noch eine jeweilige Datei, in der das eigentliche Mapping stattfindet. Im Standard finden Sie bereits für Debian und Ubuntu entsprechende *Backends* unter */etc/apt-cacher-ng*. Schauen wir uns diese für Ubuntu mal genauer an, wie in Listing 6.33 dargestellt:

```
http://de.archive.ubuntu.com/ubuntu/
```

Listing 6.33 Standardwerte: »*/etc/apt-cacher-ng/backends_ubuntu*«

Darüber wird definiert, wie die Anfragen der Clients übersetzt werden sollen. Die Standardkonfiguration ist sofort einsetzbar und muss nicht angepasst werden. Falls Sie Veränderungen vorgenommen haben, müssen Sie diese dem *apt-cacher-ng* über ein Neuladen der Konfiguration bekannt machen. Das Neuladen können Sie mittels `systemctl reload apt-cacher-ng` durchführen.

6.3.4 Clientkonfiguration

Für die Clientkonfiguration muss nur eine Datei editiert werden. Passen Sie die Konfigurationsdatei `/etc/apt/apt.conf.d/01proxy` an, ändern Sie diese, oder erstellen Sie sie, wie in Listing 6.34 dargestellt – da kein Dienst im Hintergrund läuft, ist keine weitere Aktion notwendig. Alle zukünftigen Downloads werden nun über den `apt-cacher-ng` geladen:

```
Acquire::http::Proxy "http://<HOSTNAME oder IP des CACHE>:3142";
```

Listing 6.34 Clientkonfiguration: »01proxy«

6.3.5 Fütterungszeit – bereits geladene Pakete dem Cache hinzufügen

Falls Sie den `apt-cacher-ng` auf einem System betreiben, das bereits Updates geladen hat, können Sie diese Ihrem frischen Cache hinzufügen. So können Sie bereits die erste Bandbreitensparnis generieren. Dafür müssen folgende Schritte durchlaufen werden:

1. Lokale Proxy-Konfiguration

Zunächst müssen Sie dem Server mitteilen, dass er ebenfalls über den `apt-cacher-ng` Updates laden soll (siehe Abschnitt 6.3.4, »Clientkonfiguration«).

2. Paketlisten laden

Laden Sie nun die aktuellen Paketlisten herunter. Darüber werden dem `apt-cacher-ng` die bereits vorhandenen Pakete bekannt gemacht:

```
apt update
```

3. Aufräumen der Pakete

Nun sollten Sie zunächst die benötigten Pakete von Ihrem Server entfernen. Nutzen Sie dafür folgenden Befehl:

```
apt autoclean
```

4. Verzeichnis »_import« anlegen

Erstellen Sie das Verzeichnis `_import` mit dem folgenden Befehl:

```
mkdir -p /var/cache/apt-cacher-ng/_import
```

5. Rechte anpassen

Passen Sie die Rechte an dem soeben erstellten Verzeichnis an, damit der `apt-cacher-ng` dieses auch benutzen kann. Setzen Sie dafür den nachstehenden Befehl ab:

```
chown apt-cacher-ng /var/cache/apt-cacher-ng/_import
```

6. Kopieren der Pakete

Kopieren Sie nun die lokalen Pakete mit folgendem Befehl in das Importverzeichnis:

```
cp -al /var/cache/apt/archives/* /var/cache/apt-cacher-ng/_import/
```

7. Importvorgang starten

Sie können den Importvorgang über den Button `START IMPORT` auf dem Webinterface von `apt-cacher-ng` (<http://localhost:3142/acng-report.html>) starten (siehe Abbildung 6.3).

8. Aufräumen

Nach dem erfolgreichen Import wird das Verzeichnis `_import` nicht länger benötigt. Entfernen Sie es mit folgendem Befehl:

```
rm -rf /var/cache/apt-cacher-ng/_import
```



Abbildung 6.3 Import über das Webinterface

Herzlichen Glückwunsch, Sie haben soeben die erste Bandbreitensparnis erzeugt! Diesen Vorgang müssen Sie nur einmal umsetzen. Sobald Ihre Server den `apt-cacher-ng` verwenden, füllt sich der Cache automatisch.

6.3.6 Details: Report-HTML

Wenn der Dienst gestartet ist, können Sie über die URL `http://<IP>:3142/acng-report.html` Statusinformationen abrufen, wie in Abbildung 6.4 dargestellt. Beachten Sie dabei, dass Sie `<IP>` durch die IP-Adresse Ihres Servers ersetzen müssen. Auf der Webseite können Sie sogar kleine Konfigurationen vornehmen. Einen Blick ist sie allemal wert.

Log analysis						
Period	Cache efficiency					
	Requests			Data		
	Hits	Misses	Total	Hits	Misses	Total
2020-12-28 08:50 - 2020-12-29 08:50	2 (25.00%)	6 (75.00%)	8	0.00 MiB (0.14%)	0.62 MiB (99.86%)	0.62 MiB

Note: data table is created based on the current log file. Deviation from real request count is possible due to previous log file optimization.

Abbildung 6.4 Report-HTML: »apt-cacher-ng«

6.3.7 rpm-basierte Distributionen

Leider gibt es für `rpm`-basierte Distributionen keine angepasste Software, die einen Repository-Cache bereitstellt. Selbstverständlich können Sie einen Squid oder Nginx dafür missbrauchen. Diesen fehlen allerdings die notwendigen Fähigkeiten, um mit Paketen anständig umzugehen. Daher empfehlen wir Ihnen, darauf zu verzichten.

6.4 Alles meins: Mirror

Im Gegensatz zum Proxy oder Cache wird bei einem Mirror das gesamte Repository gespiegelt. Dadurch können Sie ein vollständiges System installieren, ohne auch nur ein Paket aus dem Internet zu laden. Allerdings benötigt ein vollständiger Spiegel auch sehr viel Plattenplatz. Pro Sektion können schon mal 20 GB zusammenkommen.

6.4.1 deb-basierte Distributionen: debmirror

Installieren Sie folgende Pakete:

- ▶ *debmirror*
- ▶ *ubuntu-keyring* (auf Ubuntu) oder *debian-keyring* (auf Debian)
- ▶ *apache2*

Die benötigten Abhängigkeiten werden automatisch mit installiert. Im Beispiel verwenden wir ein Ubuntu-System, passen Sie daher gegebenenfalls die Befehle Ihrer Umgebung an.

6.4.2 Konfiguration

Da *debmirror* kein eigener Dienst ist, sondern nur ein Programm zur Verfügung stellt, müssen Sie die Struktur selbst aufbauen. Dazu gehören folgende Arbeitsschritte:

1. Benutzer und Gruppe anlegen
2. Verzeichnisstruktur anlegen
3. Mirror-Skript erstellen (Ubuntu)
4. Cronjobs einrichten
5. Schlüssel importieren
6. Mirror erstellen
7. Mirror verfügbar machen – Webdienst konfigurieren
8. Clientkonfiguration

6.4.3 Benutzer und Gruppe anlegen

Erstellen Sie zunächst einen eigenen Benutzer, der für den Spiegel zuständig ist. Das hat den Vorteil, dass die Erzeugung des Spiegels nicht mit root-Rechten durchgeführt werden muss.

```
daniel@server:/# sudo groupadd mirror
daniel@server:/# sudo useradd -g mirror -d /var/www/mirror -s /bin/bash -m \
-c "Ubuntu Mirror" mirror
```

Listing 6.35 Benutzer und Gruppe »mirror« anlegen

In Listing 6.35 wird zunächst die Gruppe *mirror* angelegt. Anschließend wird der Benutzer *mirror* mit `useradd` erzeugt. Die Parameter haben dabei nachstehende Bedeutung:

- ▶ `-g` = weist den Benutzer einer Gruppe zu, hier *mirror*.
- ▶ `-d` = Das Homeverzeichnis des Benutzers wird auf die nachstehende Pfadangabe geändert, hier `/var/www/mirror`. Da der Benutzer nur als Dienstgeber fungiert, sollte der Spiegel auch unter `/var/` liegen und nicht unter `/home/`.
- ▶ `-s` = setzt die Shell des Benutzers auf die Bash.
- ▶ `-m` = Falls das angegebene Homeverzeichnis nicht existiert, wird es mit angelegt.
- ▶ `-c` = Der Benutzer wird mit einem Kommentar versehen, hier *Ubuntu Mirror*.
- ▶ `mirror` = gibt den Benutzernamen an, der erzeugt wird.

6.4.4 Verzeichnisstruktur anlegen

Nun können wir die Verzeichnisstruktur anlegen. Dazu wechseln wir mittels `su` zum neu angelegten Benutzer und erstellen die benötigten Verzeichnisse unter seinem Namen, wie es Listing 6.36 darstellt:

```
daniel@server:/# sudo su - mirror
mirror@server:/# mkdir -p ~/bin ~/ubuntu ~/ubuntu-security ~/ubuntu-updates \
~/ubuntu-backports
```

Listing 6.36 Verzeichnisse anlegen

Da die Verzeichnisse direkt als Benutzer *mirror* erzeugt wurden, sind die Rechte bereits korrekt gesetzt.

6.4.5 Mirror-Skript erstellen (Ubuntu)

Damit der Spiegel erstellt und regelmäßig aktualisiert wird, empfiehlt sich der Einsatz eines Skripts, das über den `cron` zyklisch gestartet wird. In unserem Beispiel erzeugen wir Skripte für das Spiegeln von Ubuntu. Dazu erstellen wir ein zentrales Skript, das mit einem Parameter aufgerufen werden muss. Der Parameter gibt das zu spiegelnde Repository an. Erstellen Sie also zunächst die Datei `/var/www/mirror/bin/mirror-ubuntu.sh` mit dem Inhalt aus Listing 6.37, und geben Sie dem Skript mit `chmod +x mirror-ubuntu.sh` Ausführungsrechte:

```
1: #!/bin/bash
2: #
3: # Mirror Ubuntu-Repositorys
4: #
5: # VARS
6: # $1 sets the repository name
```

```
7:
8: REPO=$1
9: if [ -z $REPO ] ; then
10: logger -t mirror-$REPO[$$] ERROR no repository name given!
11: exit 1
12: fi
13:
14: logger -t mirror-$REPO[$$] Updating $REPO
15:
16: debmirror \
17: --passive \
18: --progress \
19: --nosource \
20: --host=de.archive.ubuntu.com \
21: -e http \
22: --root=ubuntu \
23: --dist=focal,bionic \
24: --section=multiverse,universe,restricted,main \
25: --arch=i386,amd64 \
26: --verbose \
27: /var/www/mirror/$REPO
28:
29: logger -t mirror-$REPO[$$] Finished Updating Ubuntu
```

Listing 6.37 Skript zum Laden der Spiegel: »mirror-ubuntu.sh«

Da dieses Skript zugegebenermaßen doch etwas umfangreicher ist, schauen wir uns die einzelnen Zeilen nun im Detail an:

- ▶ **Zeile 1**
Der *Shebang* des Skripts gibt an, mit welcher Sprache es interpretiert werden soll.
- ▶ **Zeile 2–7**
Kommentare, die das Skript und dessen Variablen beschreiben
- ▶ **Zeile 8**
weist der Variablen `REPO` den ersten übergebenen Parameter (`$1`) zu.
- ▶ **Zeile 9–12**
prüft, ob ein Repository-Name übergeben wurde. Wenn nicht, wird mit einer Fehlermeldung abgebrochen.
- ▶ **Zeile 14**
protokolliert im *Syslog*, dass das Update begonnen hat.
- ▶ **Zeile 16–27**
führt das Update aus (die Details erläutern wir im Anschluss).

▶ **Zeile 29**

protokolliert im *Syslog*, dass das Update beendet wurde.

Der in Zeile 16–27 von Listing 6.37 dargestellte Aufruf von `debmirror` enthält viele Parameter, die wir uns nun genauer ansehen:

- ▶ `---passive`
setzt den Download-Modus von FTP auf passiv.
- ▶ `---progress`
stellt einen Fortschrittsbalken beim Download dar. Dies ist nur relevant, wenn Sie das Skript von Hand auf einer Konsole ausführen.
- ▶ `---nosource`
gibt an, dass keine Quellen geladen werden sollen, sondern nur Binärpakete.
- ▶ `---host`
gibt den Server an, von dem der Spiegel geladen werden soll. Wählen Sie stets einen Server, der eine gute Verfügbarkeit hat und in Ihrer Nähe ist. Im Beispiel wurde *de.archive.ubuntu.com* verwendet, was eine gute Wahl ist, da sich dahinter stets mehrere verfügbare und vollständige Repositories verbergen.
- ▶ `-e`
gibt das Übertragungsprotokoll an. Im Beispiel haben wir *http* verwendet.
- ▶ `---root`
setzt das Wurzelverzeichnis auf dem Quellserver, in dem die Ubuntu-Archive liegen. Wenn nicht anders angegeben, wird im Standard stets *ubuntu* verwendet.
- ▶ `---dist`
spezifiziert durch Komma getrennt, welche Distributionen geladen werden sollen. Das Beispielskript in Listing 6.37 lädt die LTS-Versionen Focal und Bionic. Falls Sie ausschließlich die Version 16.04 im Einsatz haben, können Sie den Parameter entsprechend anpassen.
- ▶ `---section`
gibt die Sektionen an, die geladen werden sollen.
- ▶ `---arch`
spezifiziert durch Komma getrennt die Architektur. Falls Sie nur 64-Bit-Systeme betreiben, genügt es auch, nur diese Pakete zu spiegeln.
- ▶ `---verbose`
erweitert die Ausgabe.
- ▶ `/var/www/mirror/$REPO`
gibt den lokalen Speicherort des jeweiligen Spiegels an. Über den Parameter `$REPO` wird also auch das Verzeichnis angegeben. Soll zum Beispiel das *ubuntu-security*-Repository gespiegelt werden, dann finden Sie die Daten unter */var/www/mirror/ubuntu-security*.

**Bei wenig Plattenplatz: precleanup**

Falls Ihr Spiegelsystem nicht genügend Plattenplatz hat, können Sie dem Programm `debmirror` den Parameter `--precleanup` hinzufügen. Dieser sorgt dafür, dass vor dem Download Ihr lokaler Spiegel aufgeräumt wird. Die Entwickler warnen aber davor, dass dies zu einem inkonsistenten Spiegel führen kann. Setzen Sie diesen Parameter also nur im Notfall ein.

6.4.6 Cronjobs einrichten

Damit Sie nicht ständig alle Updates von Hand anstoßen müssen, können Sie mehrere Cronjobs einrichten, die das für Sie erledigen. Vorzugsweise sollte der Spiegel in der Nacht erzeugt werden, da in der Regel dann niemand Ihre Internetleitung benötigt. Erzeugen Sie also für den Benutzer *mirror* die notwendigen Cronjobs. Dazu benutzen wir das Tool `crontab` so, wie in Listing 6.38 dargestellt:

```
daniel@server:/# sudo crontab -e -u mirror
```

Listing 6.38 Cronjobs des Benutzers »mirror« öffnen

Fügen Sie anschließend die Zeilen aus Listing 6.39 in den sich öffnenden Editor ein. Damit geben Sie an, dass jede Nacht ab 01:00 Uhr stündlich die Updates laufen sollen. Über die Parameter nach dem Skript `mirror-ubuntu.sh` definieren Sie, welche Repositorys geladen werden sollen. Falls Sie also keine Pakete aus den *Backports* benötigen, können Sie die entsprechende Zeile einfach entfernen.

```
0 1 * * * bash -l -c "/var/www/mirror/bin/mirror-ubuntu.sh ubuntu"
0 2 * * * bash -l -c "/var/www/mirror/bin/mirror-ubuntu.sh ubuntu-security"
0 3 * * * bash -l -c "/var/www/mirror/bin/mirror-ubuntu.sh ubuntu-updates"
0 4 * * * bash -l -c "/var/www/mirror/bin/mirror-ubuntu.sh ubuntu-backports"
```

Listing 6.39 Cron-Konfiguration

6.4.7 Schlüssel importieren

Damit der angelegte Benutzer *mirror* den Spiegel erstellen und aktualisieren kann, benötigt er die GPG-Schlüssel der Repositorys. Diese können wir einfach aus dem System importieren, wie in Listing 6.40 dargestellt:

```
mirror@server:~$ gpg --no-default-keyring --keyring trustedkeys.gpg \
--import /usr/share/keyrings/ubuntu-archive-keyring.gpg
gpg: directory '/var/www/mirror/.gnupg' created
gpg: keybox '/var/www/mirror/.gnupg/trustedkeys.gpg' created
gpg: key 3B4FE6ACCOB21F32: 3 signatures not checked due to missing keys
gpg: /var/www/mirror/.gnupg/trustdb.gpg: trustdb created
```

```

gpg: key 3B4FE6ACCOB21F32: public key "Ubuntu Archive Automatic Signing \
Key (2012) <ftpmaster@ubuntu.com>" imported
gpg: key D94AA3F0EFE21092: 3 signatures not checked due to missing keys
gpg: key D94AA3F0EFE21092: public key "Ubuntu CD Image Automatic Signing \
Key (2012) <cdimage@ubuntu.com>" imported
gpg: key 871920D1991BC93C: 1 signature not checked due to a missing key
gpg: key 871920D1991BC93C: public key "Ubuntu Archive Automatic Signing \
Key (2018) <ftpmaster@ubuntu.com>" imported
gpg: Total number processed: 3
gpg:             imported: 3
gpg: no ultimately trusted keys found

```

Listing 6.40 GPG-Schlüssel für »mirror« importieren

Stolperfalle: richtiger Benutzer

Achten Sie darauf, dass der Import als Benutzer *mirror* durchgeführt wird! Ansonsten kann das System die GPG-Schlüssel nicht finden und bricht mit einer Fehlermeldung ab.



6.4.8 Mirror erstellen

Um zu prüfen, ob unser Skript auch funktionstüchtig ist, können wir es einfach ausführen. Fehler werden entsprechend in der Bash ausgegeben. Falls Sie nicht direkt den Download starten wollen, können Sie temporär dem Befehl `debmirror` den zusätzlichen Parameter `--dry-run` mitgeben. Damit läuft das gesamte Update, aber ohne Pakete herunterzuladen.

6.4.9 Mirror verfügbar machen – Webdienst konfigurieren

Damit Ihr Spiegel auch von Ihren Clients benutzt werden kann, müssen Sie diese verfügbar machen. In der Regel wird hierfür als Übertragungsprotokoll HTTP eingesetzt. Falls Sie bereits einen Apache auf dem System betreiben, müssen Sie lediglich einen weiteren virtuellen Host anlegen und diesen so einrichten, wie in Listing 6.41 beschrieben:

```

<VirtualHost *:80>
  DocumentRoot /var/www/mirror/
  <Directory "/var/www/mirror/">
    Options Indexes FollowSymLinks
    AllowOverride All
    Require all granted
  </Directory>
</VirtualHost>

```

Listing 6.41 Apache-Konfiguration des Spiegels: »/etc/apache2/sites-available/mirror.conf«

Abschließend müssen Sie den neuen virtuellen Host noch aktivieren und den Apache so durchstarten, wie in Listing 6.42 dargestellt:

```
daniel@server:/# sudo a2ensite mirror
[...]
daniel@server:/# sudo systemctl reload apache2
```

Listing 6.42 Host aktivieren und Apache neu laden

6.4.10 Clientkonfiguration

Damit Ihre Server von nun an über Ihren eigenen Spiegel Updates laden, müssen Sie ihnen dieses mitteilen. Dafür müssen Sie die lokale Datei `/etc/apt/sources.list` anpassen.

Speichern Sie die aktuelle Version dieser Datei unter einem anderen Namen, und legen Sie eine neue Version mit dem Inhalt aus Listing 6.43 an:

```
deb http://<IP>/mirror/ubuntu jammy universe multiverse main restricted
deb http://<IP>/mirror/ubuntu-security jammy-security universe main \
  restricted multiverse

deb http://<IP>/mirror/ubuntu-updates jammy-updates main restricted \
  universe multiverse
deb http://<IP>/mirror/ubuntu-backports jammy-backports main restricted \
  universe multiverse
```

Listing 6.43 Clientkonfiguration: »sources.list«



Release beachten!

Beachten Sie dabei, dass Sie das für das System zutreffende Release eintragen. Also `jammy` für Ubuntu 22.04 bzw. `focal` für Ubuntu 20.04.

Ersetzen Sie `<IP>` durch die IP-Adresse oder den Hostnamen des Systems, auf dem Sie den Spiegel betreiben. Beachten Sie, dass die Zeilen aus Listing 6.43 aufgrund ihrer Länge umbrochen wurden, aber eigentlich in eine Zeile gehören. Anschließend werden alle Updates von Ihrem eigenen Mirror geladen.

6.4.11 rpm-basierte Distributionen

Um einen Spiegel einer `rpm`-basierten Distribution zu erstellen, wird kein gesondertes Softwarepaket benötigt. Die Spiegelung erfolgt einfach mittels `rsync`.

Im Beispiel werden wir einen Mirror für openSUSE einrichten – Abweichungen zu CentOS stellen wir entsprechend dar. Nachstehende Punkte müssen dafür abgearbeitet werden:

1. Benutzer und Gruppe anlegen
2. Verzeichnisstruktur anlegen
3. Mirror-Skript erstellen
4. Cronjobs einrichten
5. Mirror erstellen
6. Mirror verfügbar machen – Webdienst konfigurieren
7. Clientkonfiguration

6.4.12 Benutzer und Gruppe anlegen

Zunächst sollte ein Benutzer für den Spiegelvorgang eingerichtet werden:

```
leap:/ # groupadd mirror
leap:/ # useradd -g mirror -d /srv/pub/opensuse -m -c "Mirror User" mirror
leap:/ # chown -R mirror:mirror /srv/pub/opensuse
```

Listing 6.44 Benutzer und Gruppe »mirror« anlegen

Der erste Befehl aus Listing 6.44 legt die Gruppe *mirror* an. Im zweiten Befehl wird ein gleichnamiger Benutzer angelegt. Dieser wird direkt mit dem Parameter `-g` der Gruppe hinzugefügt. Mit dem Parameter `-d` wird das Heimatverzeichnis auf `/srv/pub/opensuse` festgelegt. Falls es nicht existiert, wird es durch die Angabe von `-m` direkt mit angelegt. Zu guter Letzt wird über den Parameter `-c` noch ein Kommentar für den Benutzer angegeben.

Abschließend werden dem angelegten Benutzer Rechte am erstellten Verzeichnis zugewiesen. Wenn Sie CentOS einsetzen, sollten Sie als Verzeichnis natürlich besser `/srv/pub/CentOS` verwenden.

6.4.13 Verzeichnisstruktur anlegen: openSUSE Leap

Der Mirror soll unter `/srv/pub/opensuse` abgelegt werden. Dafür müssen die entsprechenden Verzeichnisse erstellt und dem neu angelegten Benutzer zugeordnet werden. Setzen Sie dafür die Befehle aus Listing 6.45 ab:

```
root@leap:/# mkdir -p /srv/pub/opensuse/update
root@leap:/# chown -R mirror:mirror /srv/pub/opensuse
```

Listing 6.45 openSUSE Leap: Verzeichnisse anlegen

6.4.14 Verzeichnisstruktur anlegen: CentOS

Bei CentOS soll der Mirror unter `/srv/pub/CentOS` abgelegt werden. Dafür müssen Sie die Verzeichnisse so erstellen, wie in Listing 6.46 gezeigt:

```
[root@centos ~]# mkdir -p /srv/pub/CentOS/8-stream
[root@centos ~]# cd /srv/pub/CentOS/8-stream
[...]# mkdir -p {addons,centosplus,contrib,cr,extras,fasttrack,isos,os,updates}
[...]# for i in $(ls); do mkdir $i/{i386,x86_64}; done
[...]# chown -R mirror:mirror /srv/pub/CentOS
```

Listing 6.46 CentOS: Verzeichnisse anlegen

6.4.15 Mirror-Skript erstellen

Damit Sie nicht ständig die gleichen Befehle absetzen müssen, erstellen wir ein kleines Skript, das uns die Arbeit abnimmt. Erstellen Sie dafür mit root-Rechten die Datei `/usr/bin/mirror.sh` mit den Zeilen aus Listing 6.47:

```
#!/bin/bash
SERVER=$1
MIRROR=$2
if [ -z "$1" ] ; then echo "need server"; exit 1; fi
if [ -z "$2" ] ; then echo "need mirror path"; exit 1; fi
mkdir -p $MIRROR 2>&1
echo -e "\nStarting download process..."
rsync -av --delete --progress --delay-updates --timeout=300 ${SERVER} ${MIRROR}
```

Listing 6.47 Mirror-Skript: `»/usr/bin/mirror.sh«`

Das Skript erwartet zwei Parameter: als ersten den Server, von dem die Pakete geladen werden, und als zweiten den Pfad zu Ihrem lokalen Spiegel.

Das Skript prüft zunächst, ob die benötigten Parameter übergeben wurden. Anschließend wird, falls noch nicht vorhanden, der lokale Pfad angelegt. Zuletzt startet das Skript den Synchronisierungsprozess mit `rsync`. Dies sehen wir uns nun genauer an:

- ▶ `-av`
aktiviert den *archive*- und *verbose*-Modus. Dabei wird sichergestellt, dass symbolische Links sowie die Attribute und Rechte der Ursprungsdateien erhalten bleiben. Gleichzeitig erhalten Sie eine umfangreichere Ausgabe.
- ▶ `--delete`
Hierüber wird `rsync` angewiesen, unbenötigte Dateien zu entfernen.
- ▶ `--progress`
zeigt einen Fortschrittsbalken des Vorgangs an.
- ▶ `--delay-updates`
stellt sicher, dass die neuen Dateien erst nach dem Ende der Verarbeitung in das Zielverzeichnis kopiert werden – so ist sichergestellt, dass keine Inkonsistenzen während des Herunterladens entstehen.

- ▶ `--timeout 300`
setzt die maximale Zeitspanne, in der auf eine Transaktion gewartet wird, auf 300 Sekunden – also fünf Minuten.

Damit das Skript ausgeführt werden kann, müssen dessen Rechte angepasst werden. Führen Sie dazu abschließend den Befehl aus Listing 6.48 aus:

```
daniel@server:~> sudo chmod +x /usr/bin/mirror.sh
```

Listing 6.48 Rechtekorrektur am Mirror-Skript

6

6.4.16 Cronjobs einrichten

Zur automatisierten Aktualisierung des Spiegels wird das soeben erstellte Skript nun in Cronjobs gepackt. Öffnen Sie den Cron-Editor für den Benutzer *mirror* mit dem Befehl aus Listing 6.49:

```
daniel@server:~> sudo crontab -u mirror -e
```

Listing 6.49 Cronjobs des Benutzers »mirror« bearbeiten

Fügen Sie nun im sich öffnenden Editor die Zeilen aus Listing 6.50 ein. Beachten Sie, dass die Befehle umbrochen wurden und eigentlich in einer Zeile stehen.

```
0 1 * * *    sleep $((($RANDOM/64)); /usr/bin/mirror.sh \
            "ftp.gwdg.de::pub/opensuse/distribution/leap/15.4/repo/oss/" \
            "/srv/pub/opensuse/leap/15.4/oss/"
0 3 * * *    sleep $((($RANDOM/64)); /usr/bin/mirror.sh \
            "ftp.gwdg.de::pub/opensuse/distribution/leap/15.4/repo/non-oss/" \
            "/srv/pub/opensuse/leap/15.4/non-oss/"
```

Listing 6.50 Cronjobs definieren für openSUSE Leap

Jede Nacht wird um 01:00 Uhr der Spiegel *oss* und um 03:00 Uhr der Spiegel *non-oss* geladen. Damit dieser Vorgang nicht immer strikt zu den angegebenen Uhrzeiten gestartet wird, wurde mithilfe des `sleep`-Kommandos eine Variabilität hinzugefügt. Der Befehl bewirkt, dass immer eine zufällige Anzahl an Sekunden gewartet wird, bevor der eigentliche Job gestartet wird. Wie bereits erörtert wurde, erwartet das *mirror.sh*-Skript zwei Parameter. Im ersten Parameter steht der Server, von dem der Spiegel geladen wird. Wie Sie Listing 6.50 entnehmen können, wird der *gwdg.de*-Mirror-Server verwendet. Die Gemeinschaftseinrichtung der Universität Göttingen und der Max-Planck-Gesellschaft ist einer der bekanntesten openSUSE-Mirror-Anbieter. Der Server wurde in der `Rsync`-Syntax angegeben (zwei aufeinanderfolgende Doppelpunkte zur Trennung von Server und Pfad). Im zweiten Parameter wird der lokale Pfad angegeben, den wir zuvor erstellt haben. Sie können den Serverpfad auch im Browser eingeben, um so einsehen zu können, welche Distributionen dort angeboten werden. Unter `ftp://ftp.gwdg.de/pub/opensuse` wird Ihnen der Inhalt angezeigt.



Anpassungen pro Spiegel

Beachten Sie, dass im Beispiel lediglich ein Spiegel von *openSUSE Leap 15.4* geladen wird. Wollen Sie eine andere oder mehrere Distributionen anbieten, müssen Sie die Pfade anpassen oder mehrere *mirror.sh*-Skriptzeilen erstellen!

Um einen CentOS-Mirror zu erstellen, muss ein Cronjob eingerichtet werden. Listing 6.51 zeigt ein Beispiel:

```
0 1 * * * sleep $((($RANDOM/64)); /usr/bin/mirror.sh \
  "rsync.uni-bayreuth.de::centos-stream/9-stream/" "/srv/pub/CentOS/9-stream/"
```

Listing 6.51 Cronjobs definieren für CentOS

6.4.17 Mirror erstellen

Selbstverständlich müssen Sie jetzt nicht eine Nacht warten, um zu sehen, ob der Mirror erstellt wird. Das Skript können Sie auch von der Konsole aus starten. Damit die Dateiberechtigungen korrekt gesetzt werden, sollten Sie das Skript als Benutzer *mirror* starten. Listing 6.52 zeigt die Ausgabe, die das Skript erzeugt:

```
daniel@leap:~> sudo su mirror /usr/bin/mirror.sh \
"ftp.gwdg.de::pub/opensuse/distribution/leap/15.4/repo/non-oss/" \
"/srv/pub/opensuse/leap/15.4/non-oss/"
```

Starting download process...

Welcome to ftp.gwdg.de

receiving file list ... 87 files to consider

./

.current.txt

14 100% 13.67kB/s 0:00:00 (xfr#1, to-chk=85/87)

ARCHIVES.gz

81,916 100% 159.18kB/s 0:00:00 (xfr#2, to-chk=84/87)

[...]

Listing 6.52 Auszug: Ausgabe des Skripts »mirror.sh« auf openSUSE Leap

6.4.18 Mirror verfügbar machen – Webdienst konfigurieren

Damit Ihre Clients den erstellten Spiegel auch nutzen können, müssen Sie ihn verfügbar machen. Hierfür genügt eine Webserverinstallation – mehr zum Thema erfahren Sie in Kapitel 8, »Webserver«. Im Beispiel verwenden wir einen Apache-Webserver.

Richten Sie für den Mirror einen neuen *VirtualHost* ein. Dafür erstellen Sie die Datei `/etc/apache2/vhosts.d/mirror.example.com.conf` mit dem Inhalt aus Listing 6.53:

```
<VirtualHost *:80>
    ServerAdmin admin@example.com
    ServerName mirror.example.com

    DocumentRoot "/srv/pub/opensuse"

    <Directory "/srv/pub/opensuse">
        Options FollowSymLinks Indexes
        IndexOptions FancyIndexing VersionSort NameWidth=* Charset=UTF-8 \
            TrackModified FoldersFirst XHTML
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

Listing 6.53 Konfiguration des VirtualHost: »`/etc/apache2/vhosts.d/mirror.example.com.conf`«

Selbstverständlich müssen Sie den Dateinamen und die Direktive `ServerName` Ihrer Umgebung anpassen. Nach den Änderungen müssen Sie den Dienst neu starten. Anschließend können Ihre Clients – vorausgesetzt, das `mirror.sh`-Skript ist bereits einmal vollständig gelaufen – Ihren neu geschaffenen Mirror-Server einsetzen.

6.4.19 Clientkonfiguration: openSUSE Leap

Wir bleiben im Suse-Universum und zeigen Ihnen in diesem Abschnitt, wie Sie den Clients in Ihrem Netz den neu geschaffenen Mirror-Server mitteilen. Öffnen Sie dafür den `yast2`, und wählen Sie unter `SOFTWARE` den Unterpunkt `SOFTWARE REPOSITORIES` aus. Im sich öffnenden Dialog können Sie über die Schaltfläche `ADD` oder die Tastenkombination `[Alt]+[A]` ein neues Repository hinzufügen.

Wählen Sie im folgenden Dialog als Medien-Typ »HTTP...« aus, und klicken Sie auf die Schaltfläche `NEXT`. Nun werden Sie aufgefordert, einen Namen (*Repository Name*) und die URL anzugeben. Geben Sie hier den Namen und den Pfad Ihres Spiegelservers an, zum Beispiel `http://mirror.example.com/opensuse/15.4`.

Anschließend wird Ihr Spiegelserver durchsucht und analysiert. Nach Abschluss der Verarbeitung steht Ihnen der neue Server in der Liste der Software-Repositories zur Verfügung. Damit der Server benutzt wird, müssen Sie diesen aktivieren und ihm eine niedrigere Priorität zuweisen. Alternativ können Sie auch die bisherigen Server deaktivieren oder entfernen.

6.4.20 Clientkonfiguration: CentOS

Damit CentOS-Systeme Ihren Mirror verwenden, müssen Sie die Datei *CentOS-Base.repo* im Verzeichnis */etc/yum.repos.d/* so anpassen, wie in Listing 6.54 dargestellt ist:

```
[base]
name=CentOS-$releasever - Base
baseurl=http://mirror.example.com/CentOS/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial

#released updates
[updates]
name=CentOS-$releasever - Updates
baseurl=http://mirror.example.com/CentOS/$releasever/updates/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras
baseurl=http://mirror.example.com/CentOS/$releasever/extras/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus
baseurl=http://mirror.example.com/CentOS/$releasever/centosplus/$basearch/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-centosofficial
```

Listing 6.54 Anpassungen in »/etc/yum.repos.d/CentOS-Base.repo«

Beachten Sie, dass Sie zwingend die Direktiven *mirrorlist* entfernen und die URL (fett hervorgehoben) Ihren Gegebenheiten anpassen müssen.

Kapitel 7

Backup und Recovery

Immer wieder wird in den verschiedensten Veröffentlichungen darauf hingewiesen, dass es wichtig ist, ein gutes Backup im System einzusetzen. Das Thema Recovery wird dann meist nur am Rande angeschnitten. In diesem Kapitel wollen wir Ihnen einige gute Tipps geben, wie Sie mit einfachen Mitteln ein effektives Backup erstellen können. Sie erfahren außerdem, was der Unterschied zwischen einem einfachen Backup und dem Recovery ist. Für das Thema Recovery werden wir Ihnen ein Open-Source-Tool vorstellen: »ReaR«. Mit seiner Hilfe können Sie auf einfache und effektive Weise Ihre Systeme für den Fall eines Totalausfalls sichern und in kurzer Zeit wieder genau rekonstruieren.

7

Inzwischen hat es sich glücklicherweise herumgesprochen, dass Backups nicht nur sinnvoll, sondern auch notwendig sind. Leider müssen wir aber immer wieder feststellen, dass die Backupstrategien unausgegoren sind oder nur halbherzig umgesetzt werden.

Auch teure, kommerzielle Lösungen bewahren Sie nicht zwangsläufig vor dem Daten-GAU. In diesem Kapitel zeigen wir Ihnen, worauf Sie achten sollten und wie Sie zu einer Lösung zur *Disaster Recovery* kommen, die Sie hoffentlich nie in Anspruch nehmen müssen. Dessen ungeachtet sollten Sie eine haben. Denn schließlich fährt auch niemand ohne Haftpflichtversicherung Auto.

7.1 Backup gleich Disaster Recovery?

Dass man Backup und Disaster Recovery nicht verwechseln sollte, merken viele Administratoren erst, wenn es zu spät ist: dann, wenn es wirklich darum geht, einen Server nach einem Disaster-Fall von null auf hundert wiederherzustellen:

- ▶ Sind überhaupt noch funktionierende Installationsmedien da?
- ▶ Wie war die Festplatte partitioniert?
- ▶ Welche händischen Anpassungen gab es am System?

Fragen über Fragen, die das Disaster erst wirklich zum Disaster machen – und die wertvolle Zeit kosten. Disaster Recovery scheitert in der Praxis oft an drei Problemen:

1. Es gibt keine funktionierenden unzerkratzten Installationsmedien für die oft sehr betagten Betriebssysteme mehr; ohne Installationsmedium kann auch das Grundsystem nicht mehr installiert werden. Ohne funktionsfähiges Basissystem fehlt die Grundlage, um die in einem Backup gesicherten Daten zurückspielen zu können.
2. Es gibt oft keine ausreichende Dokumentation über die verwendeten Partitionen und Dateisysteme, sodass die Konfiguration des Grundsystems durch zeitraubendes Trial and Error herausgefunden und gegebenenfalls mehrmals durchgeführt werden muss.
3. Disaster Recovery ist damit zeitraubende Handarbeit. Das ist (je nach Wichtigkeit des ausgefallenen Servers) vielleicht im Einzelfall akzeptabel; sollte es jedoch zu einem gleichzeitigen Schaden an vielen Servern gekommen sein, ist ein einzelner Administrator nicht mehr in der Lage, Dutzende Server in vertretbarer Zeit manuell wiederherzustellen.

Eine Disaster-Recovery-Strategie beschreibt, wie man von einer gänzlich unbenutzbaren IT-Infrastruktur schnellstmöglich wieder zum Normalzustand kommt. Das würde im Extremfall auch das Wiederherstellen eines abgebrannten Rechenzentrums umfassen. Im ersten Teil dieses Kapitels werden wir uns daher mit klassischen Backupstrategien beschäftigen und im zweiten Teil dann mit der Notfallwiederherstellung eines gesamten Systems.

7.2 Backupstrategien

Wer sichert wann was wie womit wohin, und wie spiele ich das Backup wieder zurück? Das ist die Frage, die wir beantworten müssen, um eine vernünftige Backupstrategie zu erhalten.

Wer ist verantwortlich?

Das ist leicht beantwortet: im Zweifel Sie als Administrator!

Wann muss gesichert werden?

Der Zeitpunkt der Sicherung sollte möglichst so gewählt werden, dass der normale Betrieb nicht zu sehr beeinträchtigt wird. Je nach Backupmethode und der Art des zu sichernden Systems können die Performance-Einbußen durch ein laufendes Backup ganz erheblich sein. Insbesondere das Sichern vieler Millionen kleiner Dateien, wie es beispielsweise bei Mailservern anfällt, ist extrem I/O-lastig und stellt manches Storage-System vor Probleme. Neben dem Zeitpunkt der Sicherung muss aber auch das Intervall festgelegt werden. Bei reinen Druckservern oder DNS-Servern mit wenigen Änderungen reicht vielleicht eine wöchentliche Sicherung. Datenbank- und LDAP-Server mit sehr vielen kritischen Änderungen müssen unter Umständen sogar stündlich gesichert werden.

Was muss gesichert werden?

Alles, was notwendig ist. Es kann reichen, nur die Nutzdaten und die Konfiguration zu sichern, zum Beispiel dann, wenn die Grundinstallation und Konfiguration des Betriebssys-

tems über andere Mechanismen definiert und sicher wiederhergestellt werden kann. Beispiele für solche Mechanismen sind *Kickstart* für Red Hat, *AutoYaST* für openSUSE/Novell oder *FAI* für Debian/Ubuntu. Soll das Backup aber im Rahmen einer Disaster-Recovery-Lösung eingesetzt werden, dann brauchen wir alles, inklusive Betriebssystem.

Wie wird gesichert?

Man unterscheidet zwischen drei verschiedenen Arten des Backups:

1. Vollbackup

Bei vollständigen Backups werden immer alle Daten gesichert, unabhängig davon, ob sie seit der letzten Sicherung verändert wurden.

2. Inkrementelles Backup

Es werden nur die Daten gesichert, die sich seit der letzten inkrementellen Sicherung geändert haben.

3. Differenzielles Backup

Es werden nur die Daten gesichert, die sich seit der letzten vollständigen Sicherung geändert haben.

Die Vor- und Nachteile der verschiedenen Methoden liegen auf der Hand. Die vollständige Sicherung hat den größten Platzbedarf. Dafür muss man im Fehlerfall aber nur ein Archiv zurückspielen. Das andere Extrem ist die inkrementelle Sicherung. Der Platzverbrauch ist gering, allerdings muss im Notfall unter Umständen eine ganze Reihe von Archiven in der richtigen Reihenfolge wieder eingespielt werden. Im Verlauf des Kapitels werden wir eine Variante vorstellen, die alle Vorteile kombiniert.

Darüber hinaus muss noch geklärt werden, ob im laufenden Betrieb gesichert werden kann oder ob dazu der Dienst oder eventuell das ganze System heruntergefahren werden muss. Die erste Variante, das *Hot Backup*, ist natürlich die bequemste und wird am häufigsten eingesetzt – leider auch oft an der falschen Stelle. Beim Sichern im laufenden Betrieb kann keine Datenkonsistenz garantiert werden. Daten können während der Sicherung verändert werden.

In vielen Fällen ist das kein Problem, aber manche Dienste reagieren darauf äußerst allergisch, insbesondere Datenbanken. Das Schlimme daran ist, dass die Sicherung noch klappt und man erst beim Wiederherstellen feststellt, dass man nur Datenmüll gesichert hat. Bei problematischen Diensten muss man daher ein *Cold Backup* durchführen, das heißt den Dienst stoppen und dann eine Sicherung durchführen. So etwas ist natürlich unschön und in vielen Fällen, wo Dienste rund um die Uhr angeboten werden müssen, unmöglich.

Ein Ausweg aus diesem Dilemma sind LVM-Snapshots. Dazu muss unter den zu sichernden Daten ein LVM liegen, und man muss noch Platz in der Volume Group haben. Man kann dann einen LVM-Snapshot anlegen. Das heißt, das Dateisystem wird in einem konsistenten Zustand eingefroren und unter einer neuen Volume Group zur Verfügung gestellt. Danach

kann man dann die Daten bequem wegsichern und den Snapshot wieder auflösen. Näheres dazu finden Sie in Abschnitt 3.2, »Rein logisch: Logical Volume Manager ›LVM««.

In anderen Fällen bieten sich Cluster- oder Replikationslösungen an. Das heißt, man hat mindestens zwei Server, die den Datenbestand mit diensteigenen Protokollen untereinander replizieren. Bei solchen Setups kann dann auf einem der Server der Dienst angehalten und ein konsistentes Backup angelegt werden, und danach wird der Dienst neu gestartet. Der andere Server übernimmt während dieser Zeit die Arbeit.



Replikation ist kein Backup

Auch das Löschen von Tabellen einer Datenbank wird je nach Replikationsmechanismus mehr oder weniger in Echtzeit repliziert. Replikation dient immer der Ausfallsicherheit und/oder der Lastverteilung, aber nie als Backup.

Womit wird gesichert?

Für kleinere Setups mit wenigen Servern, bei denen auf eine zentrale Managementinstanz verzichtet werden kann, bieten sich oft selbst gebaute Lösungen mit den klassischen Linux-Bordmitteln an. Einige davon werden im Folgenden beschrieben. Für anspruchsvollere Setups gibt es eine ganze Reihe von guten Open-Source-Lösungen, wie beispielsweise *Amanda* oder *Bacula/Bareos*.

Wohin wird gesichert?

Die Wahl der Sicherungsmedien sollten Sie davon abhängig machen, was Ihnen zur Verfügung steht und welche Datenmengen Sie in welchem Zeitraum sichern möchten. Prinzipiell ist vom USB-Stick bei kleineren Datenmengen über CDs/DVDs, Festplatten, SAN/NAS bis hin zu Magnetbändern alles denkbar.

Wichtig ist lediglich, dass Sie die Backups separat vom zu sichernden Server aufbewahren. Eine Sicherung auf eine weitere interne Festplatte hilft Ihnen wenig, wenn der gesamte Rechner ausbrennt. Ebenso ungünstig sind Sicherungen auf lediglich ein einziges RW-Medium. Sollte dieses defekt sein, verlieren Sie sämtliche Sicherungen auf einen Schlag.

Wie spiele ich das Backup zurück?

Dies ist mit Abstand der wichtigste Punkt. Das Wiederherstellen des Backups muss gut dokumentiert und ausprobiert worden sein. Wenn das Zurückspielen aus irgendeinem Grund nicht klappt, sind alle Ihre Vorarbeiten umsonst gewesen. Auch müssen Sie in regelmäßigen Abständen prüfen, ob die Sicherungen bzw. das Rücksichern noch funktionieren. Wir haben schon zu oft erleben müssen, dass Backuparchive defekt waren oder Tape Librarys immer an die falsche Stelle spulen und als Folge nur Datensalat auf dem Band war. So etwas stellen Sie natürlich immer erst dann fest, wenn Sie das Backup benötigen.

7.3 Datensicherung mit tar

tar (*Tape Archiver*) kennt sicherlich jeder Administrator. Das Tool leistet hervorragende Dienste, wenn es darum geht, dateibasierte Sicherungen durchzuführen. Im einfachsten Fall reicht das folgende Kommando, um rekursiv alle Dateien und Verzeichnisse im aktuellen Verzeichnis zu sichern:

```
root@abacus:/home# tar -cf backup.tar .
```

Listing 7.1 Erstellen eines Archivs

Die Option *c* steht für *create*. Die Option *f* für *file* sorgt dafür, dass das Archiv in die nachfolgend benannte Datei geschrieben wird und nicht nach *STDOUT*. Für Informationshungrige bietet sich zusätzlich die Option *v* für *verbose* an, um zu sehen, was gerade archiviert wird.

Achtung mit Wildcards

Das Beispiel aus Listing 7.1 zur Sicherung des Homeverzeichnisses hätte mit `tar -cv backup.tar *` keine versteckten Dateien gesichert!

Die Pathname Expansion der Shell erfasst mit `*` keine Einträge, die mit einem Punkt beginnen. Eine andere Möglichkeit zur Sicherung eines gesamten Verzeichnisses ist:
`tar -C /home -cf backup.tar .`

Im folgenden Beispiel wird ein vorhandenes Archiv im aktuellen Verzeichnis entpackt:

```
root@abacus:/home# tar -xvf backup.tar
```

Listing 7.2 Archiv im aktuellen Verzeichnis entpacken

Der Parameter *x* packt das Archiv aus und schreibt den Inhalt in das aktuelle Verzeichnis, da in einem *tar*-Archiv Pfade nicht absolut, sondern relativ gespeichert werden. Um ein Archiv in einem anderen Verzeichnis zu entpacken, können Sie die Option *C*, gefolgt von einem Verzeichnisnamen, angeben:

```
root@abacus:/home# tar -xvf backup.tar -C /tmp
```

Listing 7.3 Entpacken eines Archivs in ein anderes Verzeichnis

tar selbst führt übrigens keine Komprimierung durch. Es sorgt nur dafür, dass die Daten inklusive Zeitstempel erhalten bleiben. Netterweise können Sie über *tar* aber unter anderem *gzip* (Option *z*) und *bzip2* (Option *j*) direkt ansprechen:

```
root@abacus:/home# tar -cvzf backup.tar.gz *
root@abacus:/home# tar -cvjf backup.tar.bz2 *
```

Listing 7.4 Packen mit »tar« und gleichzeitiges Komprimieren

Beim Auspacken komprimierter Archive müssen Sie die entsprechenden Optionen ebenfalls wieder mit angeben.



Bei aktuelleren Versionen von *GNU-tar* wird oft automatisch erkannt, mit welchem Programm das Archiv dekomprimiert werden soll. Man kann sich die entsprechende Option dann sparen. Der Kompatibilität und Portabilität wegen sollten Sie die Option aber dennoch setzen. Den Inhalt von Archiven können Sie mit folgendem Kommando überprüfen:

```
root@abacus:/home# tar -tvzf backup.tar.gz
```

Listing 7.5 Prüfen eines »tar«-Archivs

7.3.1 Weitere interessante Optionen für GNU-tar

- ▶ `--absolute-names, -P`
speichert die absoluten Pfade, nicht die relativen.
- ▶ `--exclude=PATTERN`
dient zum Ausschließen von Dateien und Verzeichnissen von der Sicherung. Diese Option können Sie mehrfach angeben, wie hier im Beispiel:

```
tar -cvzf backup.tar.gz -exclude=/tmp -exclude=/proc /
```

Listing 7.6 Exkludieren von Dateien

- ▶ `--dereference, -h`
sorgt dafür, dass statt symbolischer Links die Zielfile des Links archiviert wird. Im Normalfall werden symbolische Links auch als symbolische Links im Archiv gespeichert, was für Backupzwecke in der Regel auch erwünscht ist, damit am Ende das System im Ursprungszustand wiederhergestellt werden kann – mit Symlinks.
- ▶ `--one-file-system`
Diese Option sorgt dafür, dass `tar` sich nicht über Dateisystemgrenzen hinweg bewegt. Der Schalter ist dann praktisch, wenn die Dateisysteme getrennt gesichert werden sollen oder nicht aus Versehen ein NFS-Mount mitgesichert werden soll.
- ▶ `--files-from=FILE, -T / --exclude-from=FILE, -X`
gibt eine Liste von Dateien und Verzeichnissen an, die gesichert bzw. extrahiert werden sollen oder eben genau nicht gesichert werden sollen (`--exclude-from`).

Die folgenden Zeilen erzeugen eine Datei *backup-filelist*, die Sie dann für die Sicherung mit `tar` verwenden können:

```
/etc  
/home  
/srv
```

Listing 7.7 »backup-filelist«


```
root@abacus:/home# tar -cvzf backup.tar.gz --files-from=backup-filelist
```

Listing 7.8 Sicherung mit einer »include«-Datei

- ▶ `--verify`
überprüft das Archiv nach dem Schreiben. Leider funktioniert das nicht bei komprimierten Archiven.

Sichern als Superuser oder als unprivilegierter Benutzer

Das Verhalten von `tar` variiert je nachdem, ob man als Superuser oder als unprivilegierter Benutzer angemeldet ist. Als normaler Benutzer gehören entpackte Dateien automatisch dem Benutzer, und die Dateirechte können von der `umask` beeinflusst werden. Daher lohnt sich ein Blick in die Manpage, um unangenehme Nebeneffekte zu vermeiden.



7

7.3.2 Sicherung über das Netzwerk mit tar und ssh

Über einen Umweg mit `ssh` können Sie auch Sicherungen über das Netz erledigen, wie hier im Beispiel gezeigt:

```
root@abacus:/home# tar -cvzf - /daten | ssh backupserver dd of=/BACKUP/archiv.tar.gz
```

Listing 7.9 Sichern mit »tar«, »ssh« und »dd« über das Netzwerk

Alternativ dazu können Sie auch die folgende Form verwenden:

```
root@abacus:/home# tar -cvzf - /daten | ssh backupserver cat > /BACKUP/archiv.tar.gz
```

Listing 7.10 Sichern mit »tar« und »ssh« über das Netzwerk

In Abschnitt 7.5, »Imagesicherung mit »dd«, finden Sie weitere Informationen zu `dd`. Als Ziel-datei wird in diesem Fall »-« angegeben. Das bedeutet, dass `STDOUT` als Ziel verwendet werden soll. Über eine *Pipe* wird der Datenstrom dann über `ssh` an das `dd`-Kommando übergeben, das die Daten in eine Datei schreibt. Der Restorevorgang läuft analog:

```
ssh backupserver "dd if=BACKUP/archiv.tar.gz" | tar -xvzf - -C /daten
```

Listing 7.11 Restore über das Netzwerk

7.4 Datensynchronisation mit rsync

Das Programm `rsync` dient zum Spiegeln von Verzeichnisstrukturen und kann daher auch ausgezeichnet für Datensicherungen verwendet werden. Das Tool ist von Haus aus netzwerkfähig und kann Daten von und zu entfernten Rechnern synchronisieren. Standardmäßig wird für den Übertragungsweg `ssh` genutzt. Durch einen *Prüfsummenvergleich* kann `rsync`

ohne großen Datentransfer leicht feststellen, ob Quell- und Zieldateien identisch sind oder nicht. Dadurch werden Dateien nur übertragen, wenn sie sich geändert haben. Das Protokoll ist dabei so schlau, dass selbst geänderte Dateien nicht komplett übertragen werden, sondern nur die Teile, die sich geändert haben. Gerade bei Logfiles ist das ein großer Vorteil.

7.4.1 Lokale Datensicherung mit rsync

Am folgenden Beispiel wollen wir Ihnen deutlich machen, wie Sie ein lokales Verzeichnis spiegeln können:

```
stefan@abacus:~/test$ rsync -av ./daten /BACKUP
sending incremental file list
[...]
sent 6144923 bytes received 54 bytes 4096651.33 bytes/sec
total size is 6144000 speedup is 1.00
```

Listing 7.12 Erste Sicherung eines Verzeichnisses

Die Option *a* steht für *archive* und umfasst alle für eine Spiegelung sinnvollen Parameter; *v* steht für *verbose*. Bei einem erneuten Aufruf des Befehls mit einer geänderten Datei sieht man am *speedup*, wie viel schneller die Übertragung der geänderten Datei im Verhältnis zur Übertragung aller Daten ist:

```
stefan@abacus:~/test$ rsync -av ./daten /BACKUP
sending incremental file list
[...]
sent 1026126 bytes received 32 bytes 2052316.00 bytes/sec
total size is 6145873 speedup is 5.99
```

Listing 7.13 Erneute Sicherung eines Verzeichnisses



Bei lokalen Spiegelungen erfolgt übrigens standardmäßig keine Teilübertragung von Dateien. Anders als beim Netzwerkeinsatz werden geänderte Dateien immer komplett übertragen.

7.4.2 Synchronisieren im Netzwerk mit rsync

Die Syntax zum Kopieren im Netzwerk ähnelt der des *scp*-Kommandos. Sie stellen dazu lediglich der Quelle oder dem Ziel eine IP-Adresse bzw. einen Rechnernamen voran und trennen diese bzw. diesen durch einen Doppelpunkt vom Pfad. Optional kann zu dem Rechnernamen mit *username@* noch die Benutzerkennung mitgegeben werden, die auf dem Zielsystem genutzt werden soll:

```
stefan@abacus:~/test$ rsync -av ./daten root@backupserver:/tmp/
root@backupserver's password:
```

```

sending incremental file list
[...]
sent 5090 bytes  received 6164 bytes  3215.43 bytes/sec
total size is 6151492  speedup is 546.60

```

Listing 7.14 Synchronisieren von Daten über das Netz

7.4.3 Wichtige Optionen für rsync

Das Programm rsync kennt sehr viele verschiedene Parameter, mit deren Hilfe Sie fast alle Eventualitäten abdecken können. In der folgenden Auflistung möchten wir Ihnen die wichtigsten Parameter kurz erklären:

- ▶ `-a, --archive`
Archivierungsfunktion, eine Zusammenfassung von `-rlptgoD`
- ▶ `-v, --verbose`
ausführliche Ausgabe; kann mehrfach angegeben werden, um den Detailgrad der Ausgabe zu erhöhen
- ▶ `-r, --recursive`
rekursives Kopieren aller Unterverzeichnisse mit Inhalt
- ▶ `-u, --update`
Neuere Dateien auf der Empfängerseite werden nicht überschrieben.
- ▶ `-l, --links`
Symbolische Links werden als symbolische Links kopiert.
- ▶ `-A, --acls`
ACLs bleiben erhalten.
- ▶ `-X, --xattrs`
Erweiterte Attribute bleiben erhalten.
- ▶ `-p, --perms`
Zugriffsrechte werden mitgesichert.
- ▶ `-o, --owner`
Der Eigentümer bleibt erhalten.
- ▶ `-g, --group`
Die Gruppenzugehörigkeit bleibt erhalten.
- ▶ `-D (--devices --specials)`
Geräte-dateien und Spezialdateien bleiben erhalten.
- ▶ `-t, --times`
Die `mtime` bleibt erhalten.

- ▶ `-x, --one-file-system`
Die Spiegelung wird auf das lokale Dateisystem beschränkt.
- ▶ `--delete`
Gelöschte Dateien in der Quelle werden auch auf dem Zielsystem entfernt.
- ▶ `--delete-excluded`
löscht auch Dateien im Zielsystem, die auf der Quelle im Nachhinein von der Synchronisation ausgeschlossen worden sind.
- ▶ `--partial`
Bei einem Verbindungsabbruch bleiben teilweise übertragene Daten erhalten. Der nächste Synchronisationsvorgang kann dann nahtlos fortgesetzt werden.
- ▶ `--numeric-ids`
Normalerweise versucht `rsync` anhand der Benutzer- und Gruppennamen die Besitzverhältnisse auf dem Ziel anzupassen. Für eine Datenmigration auf ein anderes System kann das hilfreich sein. Beim Backup auf einen zentralen Server ist es aber eher unerwünscht. Mit `--numeric-ids` wird dieses Mapping deaktiviert.
- ▶ `-z, --compress`
komprimiert die Übertragung.
- ▶ `--exclude=PATTERN, --exclude-from=FILE`
nimmt Dateien von der Sicherung aus. Mit der `from`-Variante kann eine Datei angegeben werden, die die Muster enthält (ähnlich wie bei `tar`).
- ▶ `--include=PATTERN, --include-from=FILE`
inkludiert bestimmte Dateien. Mit der `from`-Variante kann eine Datei angegeben werden, die die Muster enthält (ähnlich wie bei `tar`).
- ▶ `--bwlimit=KBPS`
limitiert die benutzte Bandbreite auf KBPS (Kilobytes pro Sekunde).

7.4.4 Backupskript für die Sicherung auf einen Wechseldatenträger

Für den einfachen Hausgebrauch bietet es sich an, USB-Festplatten als Backupmedium zu verwenden. Das folgende Skript können Sie als Grundgerüst dafür nutzen. Es enthält keinerlei Fehlerkontrolle. Sie sollten es daher vor dem Produktivbetrieb noch anpassen.

```
#!/bin/bash
# Mountpoint der externen Festplatte. Es wird davon ausgegangen,
# dass entsprechende Einträge in der /etc/fstab vorhanden sind.
MOUNTPOINT="/mnt/backup"

# Zielverzeichnis
DSTDIR="BACKUP"
```

```
# Optionen fuer rsync
OPTIONS="--archive --delete --delete-excluded --one-file-system"

# Mouneten der externen Festplatte
mount $MOUNTPOINT

# Nun folgt ein rsync-Aufruf für jedes Dateisystem
rsync $OPTIONS / $MOUNTPOINT/$DSTDIR/
rsync $OPTIONS /boot/ $MOUNTPOINT/$DSTDIR/
rsync $OPTIONS /var/ $MOUNTPOINT/$DSTDIR/
rsync $OPTIONS /srv/ $MOUNTPOINT/$DSTDIR/

# Aushaengen der externen Festplatte
umount $MOUNTPOINT
```

Listing 7.15 Skript zur Datensynchronisation mittels »rsync«

Kein FAT-Dateisystem

Damit Sie eine 1:1-Spiegelung erhalten, müssen Sie auf dem Zieldatenträger ein Linux- bzw. UNIX-Dateisystem verwenden. Ansonsten werden Zugriffszeiten, Zugriffsrechte und Spezialdateien falsch oder gar nicht gesichert.



7.4.5 Backupskript für die Sicherung auf einen Backupserver

Wenn Sie mehr als nur ein System sichern wollen, lohnt es sich, einen dedizierten Backupserver aufzusetzen. Es ist sinnvoll, das Backup von diesem Server aus anzustoßen und nicht von den zu sichernden Maschinen. Auf diesem Weg müssen Sie das Skript nur an einer Stelle pflegen. In einer DMZ wäre es andersherum auch gar nicht möglich. Darüber hinaus ist es natürlich deutlich sicherer, wenn das Backupskript auf dem Server liegt und nicht durch einen kompromittierten Backupclient missbraucht werden kann. Das folgende Skript soll mehrere Rechner sichern und die Daten auf dem Backupserver im Verzeichnis */BACKUP* ablegen. Unterhalb dieses Verzeichnisses gibt es für jeden Host ein Unterverzeichnis. Um verschiedene Sicherungen vorzuhalten, existieren darunter wiederum durchnummerierte Unterverzeichnisse (0–4). Im Verzeichnis 0 liegt jeweils das aktuellste Backup, das älteste liegt unter 4.

```
#!/bin/bash
# Zielverzeichnis
DSTDIR="/BACKUP"

# Optionen fuer rsync
OPTIONS="--archive --delete --delete-excluded --compress --numeric-ids"
```

```
# Extraoptionen, beispielsweise fuer eine Bandbreitenbeschraenkung
EXTRAOPTS="--bwlimit=5000"

# Excludeliste
EXCLUDE="--exclude=/tmp/* --exclude=/proc/*"

# Rechner, die gesichert werden sollen
CLIENTS="server1 server2 server3"

# Schleife zum Durchlaufen der Hostliste
for host in $CLIENTS ; do
    # Loeschen bzw. Rotieren der alten Sicherungen
    rm -rf $DSTDIR/$host/4
    mv $DSTDIR/$host/3 $DSTDIR/$host/4
    mv $DSTDIR/$host/2 $DSTDIR/$host/3
    mv $DSTDIR/$host/1 $DSTDIR/$host/2
    mv $DSTDIR/$host/0 $DSTDIR/$host/1
    mkdir $DSTDIR/$host/0

    # rsync-Aufruf
    rsync $OPTIONS $EXTRAOPTS $EXCLUDE $host:/ $DSTDIR/$host/0
done
```

Listing 7.16 Skript zur Sicherung mehrerer Server über das Netz

Das obige Skript dient nur zur Illustration und sollte nicht produktiv eingesetzt werden, da jegliche Fehlerkontrolle fehlt. Darüber hinaus werden die Daten fünffach gespeichert. Bei einem Backup von einem Server mit 100 GB Datenvolumen benötigen Sie so für die Sicherung 500 GB. Bei den aktuellen Festplattengrößen und -preisen ist das zwar kein Problem, aber bei vielen Servern wird es dann doch irgendwann schmerzhaft.

An dieser Stelle kommt unsere Lieblingsoption von `rsync` ins Spiel, und wir können unser zuvor gegebenes Versprechen (vollständige Backups mit dem Platzbedarf einer differenziellen Sicherung) einlösen. Die Option `--link-dest=DIR` sorgt dafür, dass `rsync` die zu kopierenden Daten auf dem Zielsystem mit dem Verzeichnis vergleicht, das wir mit `DIR` angegeben haben. Ist die Datei identisch, wird keine Kopie angelegt, sondern lediglich ein Hardlink auf die existierende Datei gesetzt. Es wird also ein neuer Verzeichniseintrag gesetzt, der auf die gleiche Datenstruktur verweist. Abgesehen von einem weiteren Verzeichniseintrag belegt dieser Vorgang keinen zusätzlichen Platz! Bei einem Backup von einem System mit 100 GB Datenvolumen, bei dem sich täglich aber nur ca. 1 GB ändert, würde unser gesamter Sicherungszyklus daher im Idealfall nur rund 104 GB umfassen. Der fertige `rsync`-Aufruf für das Skript sieht dann so aus:

```
rsync $OPTIONS $EXTRAOPTS $EXCLUDE $host:/ $DSTDIR/$host/0 \
--link-dest $DSTDIR/$host/1
```

Listing 7.17 Datensicherung mittels »rsync« in komprimierter Form

Die gezeigte Lösung hat den großen Charme, dass sie nur wenig Platz beansprucht und Sie trotzdem für die Rücksicherung auf eine *Vollsicherung* zurückgreifen können, da in jedem Sicherungsverzeichnis das gesamte Backup liegt. Darüber hinaus hat eine Spiegelung mit rsync natürlich den Vorteil, dass Sie sehr bequem auf einzelne Dateien oder Teilbereiche des Backups zugreifen können. Wenn Sie stattdessen mit komprimierten Archiven oder Magnetbändern arbeiten, gestaltet sich der Vorgang etwas aufwendiger.

Diese Backupmethode mit rsync erfreut sich schon seit langer Zeit großer Beliebtheit und wird auch im großen Stil in Rechenzentren zum Sichern von ganzen Serverfarmen eingesetzt. Natürlich sind die Skripte dafür individueller und raffinierter.

»rsync« benötigt viel Arbeitsspeicher

rsync ist für seinen Speicherhunger bekannt und baut am Anfang eine Liste der zu übertragenden Dateien auf. Das kann durchaus eine Weile dauern und bei großen Datenmengen einige Gigabyte an RAM verbrauchen. In manchen Fällen kann es daher sinnvoll sein, dass Sie die Sicherung aufteilen und nicht auf einen Schlag das gesamte Wurzelverzeichnis synchronisieren. Zusätzlich sollten Sie sehr genau auf eine vernünftige Definition der Exclude-Liste achten. Insbesondere die Sicherung von */proc* macht keinen Sinn. Bei Diensten, die in einer *chroot*-Umgebung laufen, kann das *proc*-Filesystem mehrfach auftauchen. Besonders das Sichern von */proc/kcore* ist unschön, da die Datei eine Referenz auf den Hauptspeicher ist. Wenn Ihr System also 8 GB RAM hat, werden diese 8 GB mit rsync mitgesichert. Noch schlimmer ist die Datei in einer *chroot*-Umgebung. Dort ist sie teilweise 128 TB groß, weil die korrekte Größe nicht bestimmt werden kann.



7.4.6 Verwendung von ssh für die Absicherung von rsync

Im Normalfall wird rsync über ssh getunnelt. Das heißt, bei jedem Aufruf müssen Sie das Passwort eingeben. Für einen automatisierten Aufruf per cron ist es daher noch nicht geeignet. Als Lösung für das Problem nutzen wir die Authentifizierung über das Public-Key-Verfahren mit einer leeren Passphrase. Damit aber nicht jeder, der in den Besitz des privaten Schlüssels gelangt, automatisch vollen root-Zugriff auf die Maschine erhält, müssen Sie den Key unbedingt absichern!

Zum Glück kann man den Public Key auf ein Kommando beschränken und sagen, von wo aus der Login erfolgen darf. Um den Key beispielsweise auf das rsync-Kommando von dem Rechner mit der IP-Adresse 192.168.100.1 zu beschränken, muss der Public Key in */root/.ssh/authorized_keys* folgendermaßen angepasst werden:

```
from="192.168.100.1",command="rsync --server --sender -vlogDtprz  
--delete-excluded --numeric-ids . /" ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA[...]
```

Listing 7.18 Den Public Key beschränken



Um den *ssh*-Key beschränken zu können, benötigen Sie den genauen Aufruf des Kommandos *rsync* auf der Gegenseite. Diesen erhalten Sie, wenn Sie *rsync* mit *-vv* aufrufen:

```
root@abacus:/# rsync -vv backupserver:/BACKUP/ /RESTORE/  
opening connection using: ssh backupserver rsync --server --sender  
-vvlogDtpre.ils . /BACKUP/
```

Listing 7.19 Ausgabe von »rsync -vv«

Falls Sie keinen direkten *root*-Login per *ssh* erlauben, müssen Sie dies nun zumindest für erzwungene Kommandos machen (siehe Listing 7.20). Näheres dazu finden Sie in Abschnitt 24.1, »Die SSH-Familie«.

```
# Authentication:  
PermitRootLogin forced-commands-only  
PubkeyAuthentication yes  
[...]
```

Listing 7.20 Auszug aus der Datei »/etc/ssh/sshd.conf«

7.5 Imagesicherung mit dd

Mit *dd* können Sie sehr leicht Images erzeugen oder zurückspielen. Das Tool liest Dateien oder Gerätedateien byteweise ein und schreibt sie wieder weg. Dadurch entsteht eine exakte 1:1-Kopie des Datenträgers. Das sind am Anfang die wichtigsten Optionen von *dd*:

- ▶ `if=FILE`
Steht für `input file`. Es gibt die Datei an, aus der gelesen werden soll. Wird die Option nicht angegeben, so wird von *STDIN* gelesen.
- ▶ `of=FILE`
Steht für `Output File`: die Ausgabedatei. Ohne diese Option wird nach *STDOUT* geschrieben.
- ▶ `bs=BYTES`
Mit `bs` wird bestimmt, wie viele Bytes am Stück gelesen und geschrieben werden sollen. Es können die üblichen Kürzel wie `K`, `M`, `G` etc. genutzt werden, um Blockgrößen auf Basis von Kilobyte, Megabyte und Gigabyte anzugeben. Weitere Kürzel stehen in der Manpage.
- ▶ `count=BLOCKS`
gibt an, wie viele Blöcke gelesen bzw. geschrieben werden sollen. Diese Angabe bezieht sich auch auf `bs`.

- ▶ `seek=BLOCKS`
überspringt die Anzahl der Blöcke beim Schreiben.
- ▶ `skip=BLOCKS`
überspringt die Anzahl der Blöcke beim Einlesen.

7.5.1 Sichern des Master Boot Records (MBR)

Der *Master Boot Record (MBR)* besteht aus den ersten 512 Bytes der Festplatte. Dort stehen der erste Teil des Bootloaders und die Partitionstabelle. Es macht also durchaus Sinn, dass Sie diesen Bereich sichern. Für den Fall, dass Sie sich den Bootloader oder die Partitionstabelle zerschießen, sind Sie dann bestens gerüstet.

```
root@abacus:~# dd if=/dev/sda of=sda.mbr bs=512 count=1
1+0 Datensätze ein
1+0 Datensätze aus
512 Bytes (512 B) kopiert, 0,0201355 s, 25,4 kB/s
```

Listing 7.21 Sichern des MBR mit »dd«

Das Zurückspielen erfolgt analog (siehe Listing 7.22). Die Angabe von `bs` und `count` können Sie sich allerdings sparen, da die Eingabedatei ja ohnehin nur 512 Byte groß ist.

```
root@abacus:~# dd if=sda.mbr of=/dev/sda
```

Listing 7.22 Zurücksichern des MBR mit »dd«

Das versehentliche Löschen einer Partition ist kein Beinbruch, da tatsächlich nur der MBR verändert wird, nicht aber die Daten auf der Festplatte. Solange nicht wirklich in die Partition geschrieben wurde, können Sie die Daten durch Neuanlegen der Partitionstabelle retten.



7.5.2 Die Partitionstabelle mithilfe von dd zurückspielen

Die Partitionstabelle belegt nur 64 Bytes, und Sie finden diese ab dem Byte 446 im MBR. Folgendes Kommando schreibt nur die Partitionstabelle neu und verändert keine anderen wichtigen Informationen im Master Boot Record:

```
root@abacus:~# dd if=sda.mbr of=/dev/sda bs=1 count=64 skip=446 seek=446
```

Listing 7.23 Zurückspielen der Partitionstabelle mit »dd«

Deutlich bequemer geht das Ganze mit `sfdisk`. Allerdings ist das Kommando auf älteren Systemen nicht immer serienmäßig installiert:

```
root@abacus:~# sfdisk -d /dev/sda > sda.ptable
```

Listing 7.24 Sichern der Partitionstabelle mit »sfdisk«



```
root@abacus:~# sfdisk /dev/sda < sda.phtable
```

Listing 7.25 Zurückspielen der Partitionstabelle mit »sfdisk«

7.5.3 Images mit dd erstellen

Natürlich können Sie auch ganze Festplatten oder Partitionen mit `dd` sichern. Das Ergebnis ist aber nur dann brauchbar, wenn Sie die Sicherung als *Cold Backup* durchführen. Die Partition darf nicht gemountet sein, um Inkonsistenzen zu vermeiden. Am leichtesten lässt sich das erreichen, indem Sie von einer beliebigen Linux-CD booten und dann die Sicherung mit `dd` durchführen. Wenn Sie eine externe Festplatte unter `/mnt/BACKUP` eingebunden haben, kann das so aussehen wie in Listing 7.26:

```
root@abacus:~# dd if=/dev/sda of=/mnt/BACKUP/sda.img bs=1024
```

Listing 7.26 Ein Image mit »dd« erstellen

Auf die Angabe von `count` können Sie in diesem Fall verzichten, da ja die gesamte erste Festplatte inklusive aller Daten abgezogen werden soll. Eine solche Sicherung ist sehr schnell, da `dd` die Daten blockweise von der Festplatte liest und sich nicht um einzelne Verzeichnisse und Dateien kümmern muss. Der Nachteil ist allerdings, dass das Image genauso groß wird wie die Festplatte, unabhängig von der tatsächlichen Belegung. Etwas platzsparender wird es, wenn Sie das Image gleich komprimieren:

```
root@abacus:~# dd if=/dev/sda bs=1024 | gzip > /mnt/BACKUP/sda.img.gz
```

Listing 7.27 Ein Image erstellen und direkt komprimieren

Die Rücksicherung sieht so aus:

```
root@abacus:~# gunzip -c /mnt/BACKUP/sda.img.gz | dd of=/dev/sda
```

Listing 7.28 Zurücksichern des Images mit »dd«



Falls Ihre Festplatte durch einen Defekt nicht mehr richtig ausgelesen werden kann, Sie die Daten aber unbedingt brauchen, dann kann das Tool `dd_rescue` aus dem Paket `ddrescue` gute Dienste leisten. `dd_rescue` bricht bei Lesefehlern nicht automatisch ab und überspringt defekte Sektoren. So kann man wenigstens den noch lesbaren Teil einer Festplatte retten. Das fertige Image kann dann mit etwas Glück mit einem Filesystem-Check wieder so weit gebracht werden, dass es gemountet werden kann.

7.5.4 Einzelne Dateien mit dd aus einem Image zurückspielen

Sofern Sie nicht eine ganze Festplatte, sondern nur eine Partition gesichert haben, ist das Zurückspielen ganz leicht. Mithilfe des *loop devices* lassen sich Images ganz normal mounten. In Listing 7.29 sehen Sie ein Beispiel für `/boot`:

```

root@abacus:/# dd if=/dev/sda1 of=/tmp/boot.img bs=1024
128488+1 Datensätze ein
128488+1 Datensätze aus
131572224 Bytes (132 MB) kopiert, 0,372759 s, 353 MB/s
root@abacus:/# mount -o loop /tmp/boot.img /mnt
root@abacus:/# ls /mnt
abi-2.6.31-20      initrd.img-2.6.31-20  System.map-2.6.31-22
abi-2.6.31-21      initrd.img-2.6.31-21  vmcoreinfo-2.6.31-20
abi-2.6.31-22      initrd.img-2.6.31-22  vmcoreinfo-2.6.31-21
config-2.6.31-20  lost+found            vmcoreinfo-2.6.31-22
config-2.6.31-21  memtest86+.bin        vmlinuz-2.6.31-20
config-2.6.31-22  System.map-2.6.31-20  vmlinuz-2.6.31-21
grub               System.map-2.6.31-21  vmlinuz-2.6.31-22
root@abacus:/#

```

Listing 7.29 Einzelne Dateien aus einem Partitionsimage zurücksichern

Bei Images von ganzen Festplatten gestaltet sich der Vorgang etwas schwieriger. Da sich Festplatten nicht mounten lassen, sondern nur Partitionen, müssen Sie zunächst bestimmen, an welcher Position im Image sich die gesuchte Partition befindet. Das können Sie manuell erledigen, indem Sie sich zunächst mit dem Tool `sfdisk` die *Partitionierung* des Images anzeigen lassen. Danach können Sie dann aus dem Anfang der Partition, multipliziert mit der Sektorgröße, den Offset errechnen. In diesem Fall liegt der Partitionsanfang bei Sektor 63 und die Sektorgröße beträgt 512 Bytes. Unser Offset ist damit 32.256:

```

root@abacus:/# sfdisk -luS /tmp/root.img
Festplatte /tmp/root.img: 25 Zylinder, 255 Köpfe, 63 Sektoren/Spur
Einheit = Sektoren von 512 Bytes, Zählung beginnt bei 0

Gerät      boot.  Anfang      Ende  #Sektoren  Id  System
/tmp/root.img1  *      63         257039  256977    83  Linux
/tmp/root.img2      257040  15888284   15631245  82  Linux Swap / Solaris
/tmp/root.img3      15888285  797145299  781257015  8e  Linux LVM
/tmp/root.img4      797145300  1953520064  1156374765  5  Erweiterte
root@abacus:/# mount -o loop,offset=32256 /tmp/root.img /mnt

```

Listing 7.30 Mounten eines Plattenimages

Wesentlich leichter ist der Vorgang, wenn Sie das Tool `kpartx` zu Hilfe nehmen. `kpartx` findet die Partitions Grenzen selbstständig und legt entsprechende *Mapping*-Einträge an, um die Partitionen bequem mounten zu können:

```

root@abacus:/# kpartx -l /tmp/root.img
loop0p1 : 0 256977 /dev/loop0 63
loop0p2 : 0 15631245 /dev/loop0 257040

```

```
loop0p3 : 0 781257015 /dev/loop0 15888285
loop0p4 : 0 1156374765 /dev/loop0 797145300
```

Listing 7.31 Auflisten der Partitionen mit »kpartx«

Wenn Sie die Option `-l` durch `-a` ersetzen, werden die Links angelegt. Danach können Sie die Partitionen mounten:

```
root@abacus:/# kpartx -a /tmp/root.img
root@abacus:/# mount /dev/mapper/loop0p1 /mnt
```

Listing 7.32 Mounten der Partitionen

Nach Abschluss der Recovery-Arbeiten können Sie die *Mappings* mit der Option `-d` wieder löschen. Zuvor muss die Partition natürlich wieder ausgehängt werden:

```
root@abacus:/# umount /mnt/
root@abacus:/# kpartx -d /tmp/root.img
loop deleted : /dev/loop0
```

Listing 7.33 Dismounten der Partitionen

Selbstverständlich können Sie eine Sicherung mit `dd` auch wieder per `ssh` über das Netz bewerkstelligen. Inklusive einer Komprimierung sieht das passende Kommando dann so aus wie in Listing 7.34:

```
dd if=/dev/sda bs=1024 | gzip | ssh root@backupserver "dd of=/BACKUP/sda.img.gz"
```

Listing 7.34 Sicherung mit »dd« über das Netz

7.5.5 Abschlussbemerkung zu dd

`dd` ist zum Sichern kompletter Server natürlich nur bedingt geeignet. Zum einen ist es hinderlich, dass die Partitionen für die Sicherung nicht gemountet sein dürfen, und zum anderen schränkt der Platzbedarf eines vollständigen Images die Einsatzmöglichkeiten ein. Kleinere Systemplatten lassen sich noch problemlos sichern. Gerade im Zeitalter der Virtualisierung, bei der virtuelle Festplatten mit teilweise nur 10 GB Kapazität durchgereicht werden, funktioniert das *Disk Imaging* noch ganz gut. Von einem Fileserver mit 10 TB Speicherkapazität wird allerdings niemand ernsthaft ein Image ziehen wollen.

Dessen ungeachtet hat das Tool den Vorteil, dass sich jeder Server nach dem Disaster-Fall ganz einfach 1:1 wiederherstellen lässt. In jedem Fall ist `dd` ein sehr nützliches Werkzeug und sollte jedem Administrator bekannt sein. Denn zumindest eine Kopie des MBR sollten Sie parat haben, wenn Sie mit dem Bootloader experimentieren. Wenn Sie sich intensiver mit *Disk Imaging* beschäftigen wollen, dann lohnt ein Blick auf `parted` aus dem gleichnamigen Paket. Es ist das Schweizer Taschenmesser unter den Imaging-Tools. Mit `parted` lassen sich Partitionen wiederherstellen, verschieben, in der Größe verändern und vieles mehr.

7.6 Disaster Recovery mit ReaR

Das Projekt *Relax & Recover (ReaR)* kann ein vorhandenes Linux-System so sichern (lassen), dass es sich jederzeit auf einem frisch ausgepackten Server (*bare metal*) in kürzester Zeit wiederherstellen lässt – die Initialisierung des Hardware-RAID-Controllers inklusive. Dabei ist ReaR selbst nicht für das Backup der Daten zuständig, sondern bindet dafür eine Vielzahl vorhandener Backupmechanismen ein. Einfache Methoden wie *rsync* oder *tgz*-Archive, aber auch anspruchsvolle Backuplösungen aus dem Enterprise-Bereich werden durch ReaR unterstützt: *Tivoli Storage Manager*, *SEP Sesam*, *Bacula/Bareos* und viele mehr stehen auf der Liste.

ReaR schließt damit eine wichtige Lücke, die eine Backupsoftware oft nicht schließen kann: das (automatische!) Wiederherstellen des kompletten Grundsystems, damit der Backupprozess anschließend die Daten wieder einspielen kann. Denn im Disaster-Fall beginnt das Problem ja schon vor dem Wiederherstellen der eigentlichen Nutzdaten der Festplatte: Initialisierung eines möglichen Hardware-RAID-Controllers, Partitionstabelle, Dateisysteme, Software-RAID-Konfiguration, Einrichtung des Boot-Managers – es muss ja erst einmal wieder ein lauffähiges Grundsystem mit funktionierenden Dateisystemen existieren, damit darauf dann der Client der Backupsoftware seinen Dienst verrichten kann.

Einige professionelle Backuplösungen verfügen mittlerweile auch über Disaster-Recovery-Tools, doch bleibt dabei stets das Risiko, ob deren Bootmedium auf der fraglichen Hardware überhaupt sicher funktionieren wird. Nicht selten sind diese Lösungen mit etwas komplizierteren Setups wie Software-RAID, verschlüsselten Partitionen oder ausgefalleneren Dateisystemen schnell überfordert, während ReaR als eingefleischte Linux-Lösung all diese Varianten mit links erledigt. Die Backuplösung *SEP Sesam* greift übrigens kurzerhand gleich selbst auf ReaR zurück – eine sehr löbliche Entscheidung. ReaR arbeitet wie folgt:

1. Aus dem laufenden System heraus wird ein individuelles Rescue-Image erzeugt. Es wird direkt aus den Komponenten erzeugt, die derzeit auf dem System installiert sind. Verwendet wird exakt der derzeit laufende Kernel, sodass dieser auch alle lokalen Anpassungen beinhaltet, die auf dem System gegebenenfalls vorgenommen worden sind. Das Rescue-Image wird auf dem betroffenen Server daher auch auf jeden Fall wieder booten und lauffähig sein, selbst wenn es sich um problematische Hardware handelt. Dabei dokumentiert ReaR den Zustand des Systems: Partitionstabelle, Dateisysteme, Software-RAID und vieles mehr.

Dieses nur wenige MByte große Rescue-System ist prinzipiell dauerhaft unverändert gültig und muss nur von Zeit zu Zeit neu erzeugt werden, wenn sich auf dem Grundsystem zu viel geändert hat. Es kann aus einem bootbaren USB-Stick oder einem ISO-File bestehen, aus dem eine CD-ROM gebrannt werden kann oder das per DHCP und PXE einem Server als Bootmedium zugewiesen wird.

2. In einem zweiten Schritt ruft ReaR eine beliebige von ReaR unterstützte Backupsoftware auf und überlässt es dieser, die eigentlichen Nutzdaten des Systems zu sichern. Wie und wohin die Backupsoftware die Daten sichert, interessiert ReaR nicht. Da das lauffähige Grundsystem später von ReaR sicher wiederhergestellt sein wird und das Rescue-System auch die lauffähig konfigurierte Backuplösung beinhaltet, reicht es dann, die hier gesicherten Daten ganz normal zurück ins Dateisystem zu spielen.

Im Disaster-Fall muss der Administrator das System nur noch über das individuelle Rettungs-Image booten, das alle Fähigkeiten besitzt, das System *bare metal* wiederherzustellen. ReaR initialisiert dazu Hardware und Festplatten und ruft anschließend die im Rescue-Image bereits mit enthaltene Backupsoftware auf, damit diese die eigentlichen Nutzdaten wieder zurück in die Dateisysteme kopieren kann.

Der ganze Prozess vollzieht sich dabei je nach Konfiguration vollautomatisch oder mit wenigen Tastatureingaben bei Bestätigungsabfragen; auf jeden Fall aber geht es schnell ohne Trial and Error und ist damit dafür geeignet, dass auch ein einzelner Administrator auf unzähligen Systemen zeitgleich einen Wiederherstellungsprozess anstoßen kann.

ReaR selbst ist übrigens in *bash* geschrieben, was zeigt, wie mächtig eine solche Skriptsprache sein kann, wenn nur gut in ihr programmiert wird. Wenn Sie möchten, können Sie ja mal beginnen, den ReaR-Quellcode zu lesen; das ist eine gute Übung zum Thema Shell-Programmierung (siehe Kapitel 27, »Scripting«).

7.6.1 ReaR installieren

Derzeit sind die Versionen 2.3 und 2.4 aktuell. SUSE bringt von Haus aus ReaR mit sich, für alle anderen Distributionen können Sie auf die umfangreich bereitgestellten Pakete im openSUSE Build Service zurückgreifen: <https://software.opensuse.org/package/rear>

Bei Debian wurden Teile des Pakets *syslinux* in die neu geschaffenen Pakete *isolinux* und *extlinux* ausgegliedert – installieren Sie diese Pakete gegebenenfalls manuell.

7.6.2 ReaR konfigurieren

ReaR hält alle vom Administrator zu editierenden Konfigurationsdateien in */etc/rear* vor. Zuerst liest ReaR von dort die Datei *site.conf*, die globale Einstellungen beinhalten soll, die Sie für alle Server in Ihrem Verbund festlegen wollen: beispielsweise die Einstellungen zur verwendeten Backupmethode. So können Sie die *site.conf* zentral über alle Server verteilen lassen. Anschließend liest ReaR die Datei *local.conf*, in der Sie Einstellungen definieren können, die nur für das lokale System gelten und die auch Optionen wieder überschreiben können, die Sie vorher in der *site.conf* gesetzt haben. Bei doppelten Einstellungen hat *local.conf* also das letzte Wort.

ReaR liest kurzerhand beide Dateien nacheinander ein; insofern ist es völlig egal, wo was definiert wird. Die Aufsplittung auf zwei Konfigurationsdateien dient nur dem Administrator als Hilfestellung bei der Organisation seiner Serverfarm. ReaR konfigurieren Sie in diesen Dateien durch einfache Variablenzuweisungen. Genau genommen führt ReaR diese Dateien als Shell-Skript aus, sodass Sie darin sogar mit Linux-Kommandos programmieren könnten.

Um ReaR in Betrieb zu nehmen, sollten mindestens die beiden Optionen `OUTPUT` und `BACKUP` gesetzt werden:

► `OUTPUT=ISO|PXE|USB`

Mit `OUTPUT` geben Sie an, in welchem Format das Rescue-Image erzeugt werden soll:

- `OUTPUT=ISO` erzeugt ein klassisches ISO-Image, aus dem Sie eine CD oder DVD brennen oder das Sie bei KVM-Lösungen als virtuelles CD-Laufwerk einbinden können.
- `OUTPUT=PXE` erzeugt die nötigen Dateien und Konfigurationen, damit Ihre Server das ReaR-Rescue-Image per DHCP/PXE aus dem Netz booten können.
- `OUTPUT=USB` schreibt das Rescue-Image auf einen vorher dafür formatierten USB-Stick, sodass Sie Ihr System direkt vom USB-Stick booten und recovern können.

► `BACKUP=REQUESTRESTORE|NETFS|EXTERNAL|...`

ReaR selbst unterstützt eine Vielzahl verschiedener Backupmechanismen, die für Backup oder Recovery aufgerufen werden:

– `BACKUP=REQUESTRESTORE`

Hierbei handelt es sich um die einfachste und zugleich gefährlichste Backupvariante, die es gibt. ReaR macht nämlich – nichts! Die Option `REQUESTRESTORE` ist eine Art »Null-Backup«. ReaR erzeugt das Rescue-Image und überlässt Sicherung und Wiederherstellung allein dem Administrator. Üblicherweise ist das also nicht das, was Sie erreichen wollen; aber Sie könnten damit auch krude Backupprogramme (per Hand) bedienen, die ReaR für eine vollautomatische Sicherung nicht unterstützt.

– `BACKUP=NETFS`

Hierbei werden alle Daten in einem `tgz`-Archiv auf einem Netzwerklaufwerk per NFS oder CIFS abgelegt. Andere Netzwerkdatsysteme wie beispielsweise CephFS könnten ebenso benutzt werden – am Ende ruft ReaR ja nur das `mount`-Kommando zum Einbinden des Netzwerkdatsystems auf.

ReaR kann das `tgz`-Archiv auch auf dem gleichen USB-Stick speichern, den Sie schon als Rescue-Image genutzt haben. Angesichts der heutigen Größen von USB-Sticks kann es schon interessant sein, jedem Server dauerhaft seinen individuellen Disaster-Recovery-Stick zu geben und diesen auch gleich als Backupmedium zu nutzen.

Wenn Sie `NETFS` in Verbindung mit einem Netzwerklaufwerk verwenden, müssen Sie über den Parameter `NETFS_URL` noch das Ziel der Datensicherung angeben; bei einer Sicherung auf einem USB-Stick heißt der Parameter `BACKUP_URL`:

```
BACKUP=NETFS

# Backup auf NFS-Laufwerk:
BACKUP_URL="nfs://nfserver.example.com/BACKUP"
# Backup auf CIFS-Laufwerk:
# BACKUP_URL="cifs://nfserver.example.com/BACKUP"
# Backup auf USB-Stick:
# BACKUP_URL="usb:///dev/disk/by-label/REAR-000"
```

Listing 7.35 Zielangabe bei der Verwendung von »NETFS«

– **BACKUP=EXTERNAL**

Mit EXTERNAL können Sie beliebige Kommandos oder eigene Shell-Skripte aufrufen, die die Datensicherung oder das Recovery durchführen. Welche Kommandos das sind, definieren Sie mit EXTERNAL_BACKUP bzw. EXTERNAL_RESTORE.

Damit Ihre eigenen Skripte auch in das Backupmedium integriert sind, müssen Sie ReaR über COPY_AS_IS anweisen, diese Dateien mit in das Rescue-Medium zu kopieren:

```
BACKUP=EXTERNAL
EXTERNAL_BACKUP=/usr/local/sbin/makebackup.sh
EXTERNAL_RESTORE=/usr/local/sbin/makebackup.sh
COPY_AS_IS=/usr/local/sbin
```

Listing 7.36 Eigene Shell-Skripte können bequem integriert werden.

– **BACKUP=TSM|DP|NBU|GALAXY|BACULA|BAREOS|RBME|DUPLICITY|NSR|SESAM**

Diese Methoden dienen zur Anbindung von *Tivoli Storage Manager* (BACKUP=TSM), *HP Data Protector* (BACKUP=DP), *Symantec NetBackup* (BACKUP=NBU), *Galaxy 5/6/7* (BACKUP=GALAXY), *Bacula* (BACKUP=BACULA), *Bareos* (BACKUP=BAREOS), *Rsync Backup Made Easy* (BACKUP=RBME), *Duplicity/Duply* (BACKUP=DUPLICITY), *EMC NetWorker/Legato* (BACKUP=NSR) und *SEP Sesam* (BACKUP=SESAM). Die jeweiligen Backupprogramme müssen selbst natürlich lauffähig konfiguriert sein, sodass ReaR dazu keine weiteren Konfigurationseinstellungen benötigt.

7.6.3 Aufrufparameter von ReaR

Für die Arbeit mit ReaR können Sie die folgenden Kommandos verwenden, die Sie sich über `rear help` ausgeben lassen können. Die wichtigsten Kommandos für den Einstieg sind:

- ▶ `rear dump`
zeigt Informationen zu Konfiguration und System an.
- ▶ `rear format`
formatiert ein Wechselmedium (USB-Stick) zur Benutzung mit ReaR.

- ▶ `rear mkbackup`
erzeugt das Rescue-Image und leitet einen Backupvorgang ein.
- ▶ `rear mkbackuponly`
erzeugt lediglich das Backup.
- ▶ `rear mkrescue`
erzeugt lediglich das Rescue-Image.
- ▶ `rear recover`
startet den Wiederherstellungsprozess.

Außerdem kennt ReaR einige optionale Aufrufparameter, unter anderem:

- ▶ `-s`
Simulationsmodus: ReaR zeigt an, welche Bestandteile von ReaR ausgeführt werden würden, also beispielsweise `rear -s mkrescue`.
- ▶ `-S`
Step-by-step-Modus: startet alle ReaR-Skripte nur nach vorherigem .
- ▶ `-v`
Verbose-Modus: macht ReaR gesprächiger (für den Anfang sehr empfehlenswert).

7.6.4 Der erste Testlauf

Für Ihren ersten Testlauf sollten Sie zunächst einen lokalen USB-Stick als Rescue-Medium nutzen und die Nutzdaten des Systems als tgz-Archiv auf einen bereitstehenden NFS-Server auslagern. Dazu müssen Sie den USB-Stick zunächst von ReaR formatieren lassen, wobei `/dev/sdX` für das Device Ihres USB-Sticks steht (oft: `/dev/sdb`):

```
backup:/etc/rear # rear -v format /dev/sdb
Relax-and-Recover 2.3 / 2017-12-20
Using log file: /var/log/rear/backup.log
USB device /dev/sdb must be formatted with ext2/3/4 or btrfs file system
Please type Yes to format /dev/sdb in ext3 format: Yes
Repartition /dev/sdb
Creating new ext3 filesystem on /dev/sdb1
```

Listing 7.37 »ReaR«: USB-Stick mit Backup auf dem NFS-Server

Anschließend können Sie in der Datei `/etc/rear/local.conf` die gewünschten Einstellungen vornehmen:

```
# Bootfähigen USB-Stick erzeugen
OUTPUT=USB
OUTPUT_URL="usb:///dev/disk/by-label/REAR-000"
```

```
# Backup-Daten als tar auf NFS-Server auslagern
BACKUP=NETFS
BACKUP_URL="nfs://nfsserver.example.com/BACKUP"
```

Listing 7.38 »ReaR«: USB-Stick als Rescue- und Backupmedium



Die Freigabe auf dem NFS- oder CIFS-Server muss entsprechend eingerichtet sein. Ist Ihr USB-Stick groß genug, können Sie die Backupdaten auch gleich auf den Stick sichern lassen:

```
# Bootfähigen USB-Stick erzeugen
OUTPUT=USB
OUTPUT_URL="usb:///dev/disk/by-label/REAR-000"

# Backup-Daten als tar direkt auf dem USB-Stick speichern
BACKUP=NETFS
BACKUP_URL="usb:///dev/disk/by-label/REAR-000"
```

Listing 7.39 »ReaR«: Minimalkonfiguration

Alternativ können Sie statt des Recovery-Mediums auf dem USB-Stick auch eine ISO-Datei auf dem NFS-Server ablegen lassen, die Sie in einem zweiten Schritt als Rescue-Medium auf CD brennen:

```
# ISO-Image auf dem NFS-Server erzeugen
OUTPUT=ISO

# Backup-Daten als tar auf NFS-Server auslagern
BACKUP=NETFS
BACKUP_URL="nfs://nfsserver.example.com/BACKUP"
```

Listing 7.40 ISO-Datei erzeugen

Nachdem Sie Ihre Konfiguration für ReaR vorgenommen haben, können Sie über das Kommando `rear dump` sich noch einmal alles ausgeben lassen und prüfen:

```
backup:~ # rear dump
Relax-and-Recover 2.3 / 2017-12-20
Using log file: /var/log/rear/backup.log.lockless
Dumping out configuration and system information
This is a 'Linux-x86_64' system, compatible with 'Linux-i386'.
System definition:
                ARCH = Linux-i386
                OS = GNU/Linux
                OS_MASTER_VENDOR = SUSE
                OS_MASTER_VERSION = 12
                OS_MASTER_VENDOR_ARCH = SUSE/i386
                OS_MASTER_VENDOR_VERSION = SUSE/12
```

```

OS_MASTER_VENDOR_VERSION_ARCH = SUSE/12/i386
    OS_VENDOR = SUSE_LINUX
    OS_VERSION = 12
    OS_VENDOR_ARCH = SUSE_LINUX/i386
    OS_VENDOR_VERSION = SUSE_LINUX/12
OS_VENDOR_VERSION_ARCH = SUSE_LINUX/12/i386

```

Configuration tree:

```

Linux-i386.conf : OK
GNU/Linux.conf : OK
    SUSE.conf : missing/empty
    SUSE/i386.conf : missing/empty
    SUSE/12.conf : missing/empty
    SUSE/12/i386.conf : missing/empty
    SUSE_LINUX.conf : OK
    SUSE_LINUX/i386.conf : missing/empty
    SUSE_LINUX/12.conf : missing/empty
    SUSE_LINUX/12/i386.conf : missing/empty
    site.conf : missing/empty
    local.conf : OK

```

Backup with NETFS

```

NETFS_KEEP_OLD_BACKUP_COPY =
    NETFS_PREFIX = backup
BACKUP_INTEGRITY_CHECK =
    BACKUP_MOUNTCMD =
    BACKUP_OPTIONS =
    BACKUP_RSYNC_OPTIONS = --sparse --archive
    --hard-links --numeric-ids --stats
BACKUP_SELINUX_DISABLE = 1
    BACKUP_TYPE =
    BACKUP_UMOUNTCMD =
    BACKUP_URL = usb/dev/disk/by-label/REAR-000

```

Backup program is 'tar':

```

    BACKUP_PROG = tar
    BACKUP_PROG_ARCHIVE = backup
BACKUP_PROG_COMPRESS_OPTIONS = --gzip
BACKUP_PROG_COMPRESS_SUFFIX = .gz
    BACKUP_PROG_CRYPT_ENABLED = 0
    BACKUP_PROG_CRYPT_KEY =
    BACKUP_PROG_CRYPT_OPTIONS = /usr/bin/openssl des3 -salt -k
BACKUP_PROG_DECRYPT_OPTIONS = /usr/bin/openssl des3 -d -k
    BACKUP_PROG_EXCLUDE = /tmp/* /dev/shm/*
    /var/lib/rear/output/*
    BACKUP_PROG_INCLUDE =

```

```
                BACKUP_PROG_OPTIONS = --anchored
    BACKUP_PROG_OPTIONS_CREATE_ARCHIVE =
    BACKUP_PROG_OPTIONS_RESTORE_ARCHIVE =
                BACKUP_PROG_SUFFIX = .tar
    BACKUP_PROG_WARN_PARTIAL_TRANSFER = 1
Output to USB
                USB_DEVICE =
                USB_FILES =
    USB_RETAIN_BACKUP_NR = 2
                RESULT_MAILTO =
```

Listing 7.41 Testen der Konfiguration

Die Ausgabe von `rear dump` sollten Sie mitsenden, wenn Sie einmal auf der ReaR-Mailingliste um Rat fragen wollen. Die ersten beiden Blöcke, `System definition` und `Configuration tree`, dienen nur zu Ihrer Information und um alle Details für ein schnelles Debugging zur Hand zu haben. Wundern Sie sich dabei nicht über die zahlreichen Konfigurationsdateien im `Configuration tree`. ReaR prüft hier zunächst das Vorhandensein generischer Konfigurationsdateien (»Das ist ein Linux-System«), bevor es nach distributionsspezifischen Details schaut (»Das ist ein SUSE-System«) und am Ende die tatsächlich vom Administrator angepassten individuellen Konfigurationen `/etc/rear/site.conf` und `/etc/rear/local.conf` sucht. Wenn einige dieser ReaR-eigenen Konfigurationsdateien nicht vorhanden sind (`missing/empty`), so ist das nicht schlimm oder ungewöhnlich: In diesem Fall gibt es für Ihre besondere Architektur seitens ReaR keine besondere Unterkonfiguration, und alles kann nach dem normalen Prinzip ablaufen. Interessant ist für Sie als Administrator in der Endkontrolle eigentlich nur der Teil mit den Backupereinstellungen, die Sie während der Konfiguration vorgenommen haben: Stimmt alles? Sind alle Ihre Angaben auch von ReaR so gefunden und verstanden worden?

Sieht alles richtig aus, können Sie ReaR starten und das Rescue-Image und das Backup in einem gemeinsamen Durchlauf erzeugen lassen:

```
backup:~ # rear -v mbackup
Relax-and-Recover 2.3 / 2017-12-20
Using log file: /var/log/rear/rear-T510-15.log
Creating disk layout
Creating root filesystem layout
TIP: To login as root via ssh you need to set up
/root/.ssh/authorized_keys or SSH_ROOT_PASSWORD in your configuration file
Copying files and directories
Copying binaries and libraries
Copying kernel modules
Creating initramfs
Writing MBR to /dev/sdb
Copying resulting files to usb location
```

```

Encrypting disabled
Creating tar archive
'/tmp/rear.fz668YrNdclPJqJ/outputfs/rear/T510-15/20160721.1201/backup.tar.gz'
Archived 3237 MiB [avg 6461 KiB/sec]OK
Archived 3237 MiB in 514 seconds [avg 6449 KiB/sec]

```

Listing 7.42 Der Backupprozess mit »ReaR«

Wenn alles erfolgreich funktioniert hat, sollten Ihre Daten in Form eines tgz-Archivs auf dem NFS-Laufwerk liegen und Sie – je nach Konfiguration – einen bootbaren USB-Stick besitzen bzw. ein auf CD brennbares ISO-Image auf dem NFS-Server vorfinden.

7

7.6.5 Der Recovery-Prozess

Prüfen Sie, ob der Recovery-Prozess funktioniert, und spielen Sie den Recovery-Fall einmal durch. Achtung: Da Sie das Ganze erst noch auf Fehlerfreiheit testen müssen, sollten Sie das Folgende nur auf einer Testmaschine ausführen, deren Daten im Falle eines Fehlers auch verloren gehen können. Booten Sie vom besagten USB-Stick oder von einer Rescue-CD, die Sie eingelegt haben. Achten Sie darauf, dass der Stick oder das CD-Laufwerk auch im BIOS als Bootmedium vor Ihrer eigenen Festplatte angesprochen wird.

Auch wenn Ihr Server so konfiguriert ist, dass er Ihr ReaR-Rescue-Image als erstes Bootmedium bootet, droht Ihrer Maschine keine Gefahr: Der Bootloader von ReaR ist so konfiguriert, dass er nach wenigen Sekunden ohne Tastatureingabe den normalen Bootvorgang von der Festplatte einleitet. Sie können den bootfähigen USB-Stick also angesteckt bzw. Ihre ReaR-Rescue-CD dauerhaft im CD-Laufwerk eingelegt lassen.

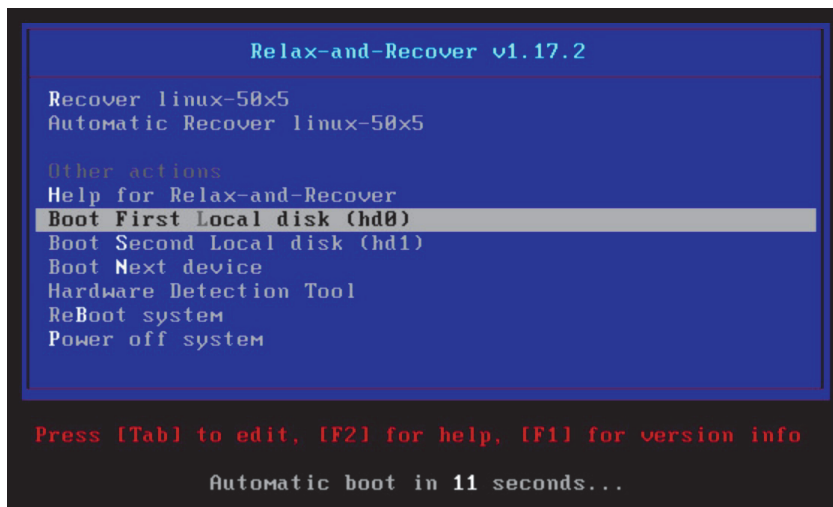


Abbildung 7.1 Der Bootprompt von »ReaR«

Nur wenn Sie innerhalb der ersten 30 Sekunden am Boot-Prompt die Entscheidung treffen, in den ReaR-Recovery-Modus zu starten, wird ReaR sein mit mkrescue angelegtes Minimal-System starten, in das Sie sich als root einloggen können (siehe Abbildung 7.1). Da Ihr gebootetes System dieselbe Netzwerkkonfiguration wie der ursprüngliche Server besitzt, können Sie sich alternativ auch per ssh in das System einloggen; die für Logins hinterlegten Public Keys des Originalservers sind im Rescue-System ebenfalls mit enthalten. Da das Rescue-System kein root-Passwort beinhaltet, ist ein Passwort-Login jedoch nicht möglich. Mit dem Kommando `rear recover` formatieren Sie nun die Festplatte und leiten damit die tatsächliche unwiderrufliche Wiederherstellung Ihres Servers ein:

```
RESCUE backup:~ # rear recover
Relax-and-Recover 2.3 / 2017-12-20
Using log file: /var/log/rear/rear-backup.log
NOTICE: Will do driver migration
Select a backup archive.
1) 20160721.1522
2) Abort
#? 1
2016-07-11 06:28:48 Backup archive /tmp/rear.cqbCkP7CmgwoE57/outputfs/rear/backup/
20160721.1522/backup.tar.gz chosen.
Backup archive /tmp/rear.cqbCkP7CmgwoE57/outputfs/rear/backup/20160721.1522/backup
.tar.gz chosen.
Calculating backup archive size
Backup archive size is 16G /tmp/rear.cqbCkP7CmgwoE57/outputfs/rear/backup/\
20160721.1522/backup.tar.gz (compressed)
Comparing disks.
Start system layout restoration.
Creating partitions for disk /dev/sda (msdos)
Creating partitions for disk /dev/sdc (msdos)
Please enter the password for cr_ata-ST9320423AS_5VJ7S8YP-part2(/dev/sda2):
Enter passphrase:
Please re-enter the password for cr_ata-ST9320423AS_5VJ7S8YP-part2(/dev/sda2):
Enter passphrase for /dev/sda2:
Enter passphrase for /dev/sda2:
Creating LVM PV /dev/mapper/cr_ata-ST9320423AS_5VJ7S8YP-part2
Creating LVM VG system
Creating LVM volume system/home
Creating LVM volume system/root
Creating LVM volume system/swap
Creating ext4-filesystem / on /dev/mapper/system-root
Mounting filesystem /
Creating ext4-filesystem /home on /dev/mapper/system-home
Mounting filesystem /home
```

```

Creating ext4-filesystem /boot on /dev/sda1
Mounting filesystem /boot
Creating ext3-filesystem /tmp/rear.TkR4JvFG0sf4L5f/outputfs on /dev/sdc1

```

Listing 7.43 System-Recovery mit »ReaR«

ReaR hat bei diesem Recovery-Prozess das RAID-System, Partitionen, den LVM und die Dateisysteme wieder angelegt und anschließend das Wiedereinspielen der gesicherten Daten veranlasst. Das dauert oft nur wenige Sekunden oder Minuten – die genaue Dauer hängt natürlich vom Volumen und von dem gewählten Backupmedium ab. Als Letztes wird der Bootloader auf der Festplatte installiert. Bevor Sie Ihr wiederhergestelltes System neu booten, können Sie auf Wunsch alles unter `/mnt/local` noch einmal prüfen.

7.6.6 Die ReaR-Konfiguration im Detail

In der Datei `/usr/share/rear/conf/default.conf` sind bereits zahlreiche Konfigurationen vordefiniert. Wenn Sie diese anpassen möchten, so überschreiben Sie diese Werte nicht in diesen Dateien, sondern übertragen die Optionen in Ihre eigene `/etc/rear/local.conf`.

► PROGS | REQUIRED_PROGS

Diese Listen (Arrays) enthalten die Namen von Programmen, die optional (PROGS) bzw. zwingend (REQUIRED_PROGS) in das Rettungssystem integriert werden sollen. Welche Programme ReaR per Default in das Rescue-Image integriert, hängt vom Betriebssystem und vom Versionsstand ab und wird von ReaR automatisch erkannt und dynamisch konfiguriert. Sie dürfen als Administrator bei eigenen Anpassungen diese Variablen also nicht komplett neu setzen, sondern stets nur wie folgt ergänzen:

```
PROGS=( "${PROGS[@]}" programm1 programm2 /usr/local/sbin/makebackup.sh )
```

Sofern die genannten Programme in den Pfaden der PATH-Variablen zu finden sind, reicht ReaR die Angabe des reinen Programmnamens. Abhängigkeiten zu verlinkten Bibliotheken werden automatisch erkannt und mit aufgelöst.

► LIBS

Möchten Sie zusätzlich zu den automatisch erkannten Bibliotheken noch weitere mit in das Rescue-Image integrieren, können Sie diese in diesem Array ergänzen.

► MODULES | MODULES_LOAD

Diese Arrays definieren die zu integrierenden Kernelmodule (MODULES) bzw. die bereits im Rettungssystem auch zu ladenden Module (MODULES_LOAD). Aber auch die notwendigen Kernelmodule können von ReaR automatisch erkannt werden.

► CLONE_USERS | CLONE_GROUPS

Aus Sicherheitsgründen enthält Ihr Rescue-System normalerweise nur eine minimalisierte Benutzerliste. Sollten Sie für Ihre Backupsoftware zusätzliche Benutzer benötigen, können Sie über diese Liste weitere zu integrierende Nutzer und Gruppen festlegen.

- ▶ `COPY_AS_IS` | `COPY_AS_IS_EXCLUDE`
definiert als Liste Dateien und Verzeichnisse, die ebenfalls in das Rescue-System integriert werden (`COPY_AS_IS`) bzw. davon ausgeschlossen werden sollen (`COPY_AS_IS_EXCLUDE`).
- ▶ `EXCLUDE_MOUNTPOINTS` | `EXCLUDE_MD` | `EXCLUDE_VG`
Möchten Sie Mountpoints, MD-Devices oder ganze LVM-Volume-Gruppen ausschließen, können Sie dies hier definieren.

7.6.7 Migrationen mit ReaR

ReaR ist dazu entwickelt worden, ein System 1:1 auf der ursprünglichen (oder einer identischen) Hardware wiederherzustellen. Doch diese Zeiten sind seit Version 1.9 vorbei: ReaR kommt mittlerweile ebenso gut mit geänderter Hardware (anderen RAID-Controllern, anderen Netzwerkinterfaces) oder sogar mit virtualisierter Hardware klar. ReaR kann darum auch als mächtiges Werkzeug für geplante Migrationen zwischen zwei Hardwareplattformen (*P2P*) oder in eine Virtualisierungsumgebung (*P2V*) genutzt werden. Genau genommen können Sie mit ReaR sogar eine Migration von der einen virtuellen Umgebung in die nächste anstoßen (*V2V*) – oder von einer virtuellen Umgebung zurück auf echte Hardware (*V2P*), was normalerweise kaum ein Migrationstool unterstützt.

```

Welcome to Relax and Recover. Run "rear recover" to restore your system !

RESCUE linux-50x5:~ # rear recover
Relax-and-Recover 1.17.2 / Git
Using log file: /var/log/rear/rear-linux-50x5.log
NOTICE: Will do driver migration
Calculating backup archive size
Backup archive size is 1.9G /tmp/rear.hb5rcufW8R6u06A/outputfs/linux-50x5/backup.tar.gz (compressed)
Comparing disks.
Device sda has size 34359738368, 17179869184 expected
Switching to manual disk layout configuration.
Original disk /dev/sda does not exist in the target system. Please choose an appropriate replacement
.
1) /dev/sda
2) Do not map disk.
#? 1
2016-07-26 09:33:33 Disk /dev/sda chosen as replacement for /dev/sda.
Disk /dev/sda chosen as replacement for /dev/sda.
This is the disk mapping table:
/dev/sda /dev/sda
Please confirm that '/var/lib/rear/layout/disklayout.conf' is as you expect.

1) View disk layout (disklayout.conf) 4) Go to Relax-and-Recover shell
2) Edit disk layout (disklayout.conf) 5) Continue recovery
3) View original disk space usage 6) Abort Relax-and-Recover
#? -

```

Abbildung 7.2 Erkennt ReaR geänderte Hardware, wird der Administrator um Freigabe gebeten.

ReaR entdeckt geänderte Hardware bei der Recovery und passt Einstellungen im System an diese Hardware an, nachdem es den Administrator um Erlaubnis gefragt hat. Geänderte MAC-Adressen oder andere Festplattensysteme – kein Problem. Da neue Hardware selten

leistungsschwächer ist als die alte, haben neue Server oft auch größeren Festplattenspeicher an Bord. Auch damit kann ReaR umgehen, indem es die Partitionstabelle auf Wunsch anpasst, um die Festplatte voll zu nutzen (siehe Abbildung 7.2).

Wenn Sie automatisiert eine größere Anzahl von Systemen migrieren wollen, wird Ihnen der interaktive Modus eher lästig sein. ReaR sollte dann nicht nach seinen neuen Einstellungen fragen, ReaR sollte sie kennen und automatisch verwenden.

Auch das ist möglich: Sie können noch auf dem alten System vor dem Erstellen des Backups die Dateien `/etc/rear/mappings/ip_addresses` und `/etc/rear/mappings/routes` mit den Daten füllen, die ReaR bei der Recovery verwenden soll, und so sogar dafür sorgen, dass das wiederhergestellte System mit einer neuen IP-Konfiguration ausgestattet wird:

```
# cat /etc/rear/mappings/ip_addresses  
eth0 192.168.77.22
```

```
# cat /etc/rear/mappings/routes  
default 192.168.77.2
```

Listing 7.44 Vordefinierte Migrationen von IP-Adressen und IP-Routen

Eine Migration auf kleinere Festplatten ist derzeit jedoch noch nicht möglich. Dieser Fall tritt eher selten auf – in der Regel wird die Migration auf größere, anstatt auf kleinere, Festplatten durchgeführt.



TEIL III
Dienste

Kapitel 8

Webserver

In diesem Kapitel erlernen Sie die Konfiguration der Webserver »Apache« und »nginx«. Die Schwerpunkte dieses Kapitels liegen auf der Einrichtung virtueller Hosts mit und ohne SSL, der Einbindung von PHP und den essentiellen Schutzmaßnahmen.

Obwohl es beileibe nicht an Herausforderern mangelt, wird ein Großteil aller Webangebote von einem *nginx*- oder *Apache*-Webserver ausgeliefert. Dem langjährigen Platzhirsch Apache wurde der Rang der meist eingesetzten Webservers im Internet in den letzten Jahre durch den Konkurrenten *nginx* streitig gemacht. Was die beiden unterscheidet und wie Sie mit ihnen erfolgreich und sicher Webseiten betreiben können, zeigen wir Ihnen in diesem Kapitel.

Dabei bilden virtuelle Hosts und HTTPS den Kern dieses Kapitels, ergänzt um Apache-Spezifika wie das Sicherheits-Plug-in *ModSecurity* und Erläuterungen zum Tuning. Ebenso zeigen wir Ihnen wie Sie das noch immer viel eingesetzte PHP sowohl mit Apache als auch mit *nginx* betreiben können. Ausführlich gehen wir auf die heutzutage essenzielle SSL/TLS-Konfiguration ein, wie Sie diese noch sicherer gestalten können und wie Sie Ihre Arbeit überprüfen können.

8.1 Apache

Selbst wenn Sie nur ein einziges Webangebot auf Ihrem Server betreiben möchten, lohnt es sich, das Angebot als virtuellen Host zu konfigurieren. Auf diese Art ist die Konfiguration übersichtlicher, und Sie können schneller reagieren, wenn eines Tages doch einmal ein zweites Webangebot (oder gar mehr) auf dem Server einziehen soll. Daher ist diese Betriebsart auch zum Standard geworden.

8.1.1 Installation

Selbstverständlich gehört der Apache zum Standard und ist in den Paketquellen vorhanden. Die Installation erfolgt daher wie gewohnt:

- ▶ **Debian:** `apt install apache2`
- ▶ **Ubuntu:** `apt install apache2`

- ▶ **CentOS:** `dnf install httpd`
- ▶ **openSUSE Leap:** `zypper install apache2`

Nach dem Installationsvorgang finden Sie die zentralen Konfigurationsdateien unter `/etc/apache2/`, außer bei CentOS. Dort lautet nicht nur der Pfad `/etc/httpd`, sondern auch der Daemon und somit auch die Start-Skripte heißen so.

Die Hauptkonfigurationsdatei heißt `httpd.conf`. Jede Distribution verfolgt einen anderen Ansatz, wie die Konfigurationen verteilt und eingebunden werden. Debian und Ubuntu arbeiten nach dem gleichen Standard, CentOS verfolgt eine ganz eigene (eher an die Ursprünge erinnernde) Philosophie, und openSUSE Leap steht irgendwo dazwischen. Im weiteren Verlauf dieses Abschnitts arbeiten wir mit einem Ubuntu-System. Daher werden wir Ihnen an der einen oder anderen Stelle die Abweichungen, zum Beispiel bei den Pfaden oder den zu verwendenden Konfigurationsdateien, entsprechend ausweisen.

8.1.2 Virtuelle Hosts einrichten

Um mehrere Webangebote auf einem Server zu betreiben, bieten sich namensbasierte virtuelle Hosts an. Der Webserver »lauscht« dabei nur auf einer einzigen IP-Adresse und entscheidet anhand der HTTP-Header in der Anfrage des Clients, welcher virtuelle Host die Anfrage beantworten soll. Standardmäßig sind bei Debian, Ubuntu und openSUSE Leap bereits zwei virtuelle Hosts eingerichtet – einer für HTTP (`000-default.conf` bzw. `vhost.template`) und einer für HTTPS (`default-ssl.conf` bzw. `vhost-ssl.template`). Im Folgenden richten wir nun einen eigenen virtuellen Host für die Webseite `example.com` ein:

```
<VirtualHost *:80>
    ServerName example.com
    ServerAlias sub.example.com www.example.com

    ServerAdmin webmaster@example.com
    DocumentRoot /var/www/example.com/
    ### openSUSE Leap: /srv/www/vhosts/example.com

    ErrorLog /var/log/apache2/example.com_error.log
    CustomLog /var/log/apache2/example.com_access.log combined
    ### CentOS: /var/log/httpd/
    <Directory /var/www/example.com>
    ### openSUSE Leap: /srv/www/vhosts/example.com
        Options Indexes FollowSymLinks

        ### openSUSE Leap (nur diese Zeilen):
        # Options Indexes FollowSymLinks
        # AllowOverride None
```

```

# <IfModule !mod_access_compat.c>
#   Require all granted
# </IfModule>
# <IfModule mod_access_compat.c>
#   Order allow,deny
#   Allow from all
# </IfModule>
###
</Directory>
</VirtualHost>

```

Listing 8.1 Konfigurationsdatei für einen virtuellen Host

Debian/Ubuntu

Bei Debian- und Ubuntu-Systemen legen Sie im Verzeichnis `/etc/apache2/sites-available/` für jeden virtuellen Host eine Konfigurationsdatei an. Für unser Beispiel sollten Sie daher die Datei `/etc/apache2/sites-available/example.com.conf` mit dem Inhalt aus Listing 8.1 anlegen. Achten Sie darauf, dass Ihre Konfigurationsdatei mit der Endung `.conf` versehen ist, da Apache nur Dateien mit dieser Endung verarbeitet.

openSUSE Leap

Auf openSUSE-Leap-Systemen müssen Sie ebenfalls eine Konfigurationsdatei für die virtuellen Hosts erstellen – dort aber im Verzeichnis `/etc/apache2/vhost.d`. Für unser Beispiel müssen Sie also die Datei `/etc/apache2/vhost.d/example.com.conf` mit dem Inhalt aus Listing 8.1 anlegen. Auch hier ist die Endung `.conf` Pflicht. Passen Sie dort die Pfade in den Direktiven `DocumentRoot` und `Directory` auf `/srv/www/` an, und entfernen Sie die Kommentarzeichen für die zusätzliche Modulabfrage, da standardmäßig keine Zugriffe erlaubt sind. Ohne die Abfrage erhalten Sie beim Aufruf die Fehlermeldung `403 – Zugriff verweigert`.

CentOS

Bei CentOS sind keine Verzeichnisse für das Beherbergen von virtuellen Hosts explizit vorgesehen. Daher können Sie die Konfigurationsdatei entweder an dem zentralen Konfigurationsort unter `/etc/httpd/conf.d` ablegen oder einen eigenen Ordner anlegen (zum Beispiel wie bei openSUSE Leap `vhost.d`) und diesen mit der Direktive `IncludeOptional vhost.d/*.conf` in `httpd.conf` einbinden.

Legen Sie nun eine Konfigurationsdatei mit dem Inhalt aus Listing 8.1 unter dem Verzeichnis Ihrer Wahl ab, und zwar unter dem Namen `example.com.conf`. Bei CentOS müssen Sie die Pfade zu den Log-Dateien entweder absolut (`/var/log/httpd/<LOG>.log`) oder relativ (`logs/<LOG>.log`) anpassen und das `DocumentRoot` und `Directory` auf `/var/www/` anpassen.

Allgemein

Die Direktiven aus Listing 8.1 haben im übrigen nachstehende Bedeutung:

- ▶ `ServerName`
Mit diesem Namen weist sich der Server dem Client gegenüber aus, auch wenn er – vom DNS gesteuert – noch unter anderen Namen erreichbar ist.
- ▶ `ServerAlias`
Hier definieren Sie Alias-Namen für Ihr Webangebot, wenn Sie möchten, dass Ihr Angebot unter mehreren Namen aufgerufen werden kann. An dieser Stelle sind auch Wildcards möglich, zum Beispiel `ServerAlias *.example.com`.
- ▶ `ServerAdmin`
Hier können Sie als Admin Ihre E-Mail-Adresse eintragen (oder, wenn Sie schlau sind, die des Helpdesks).
- ▶ `DocumentRoot`
Das ist das Wurzelverzeichnis Ihres Webangebots. Viele Tools und Hilfsprogramme verlassen sich darauf, die Verzeichnisse der virtuellen Hosts unterhalb von `/var/www` zu finden, weshalb Sie dieser Konvention folgen sollten. Bei openSUSE Leap ist das Standardverzeichnis im Übrigen unter `/srv/www` zu finden.
- ▶ `ErrorLog`
Die hier eingetragene Datei sammelt Fehlermeldungen dieses virtuellen Hosts.
- ▶ `CustomLog`
Alle Webzugriffe auf den virtuellen Host erzeugen einen Eintrag in diesem Logfile. Wenn Sie hier, wie im Beispiel, das `combined`-Format spezifizieren, erhalten Sie etwas mehr Informationen als beim klassischen Webserver-Log.
- ▶ `Directory`
Hier legen Sie mannigfaltige Optionen für den angegebenen Pfad fest. Im Beispiel sehen Sie die beiden sicherlich am häufigsten benötigten Optionen: `Indexes` zeigt eine Liste aller Dateien im Verzeichnis an, wenn keine *Index-Datei* (oft heißt sie `index.html`, `home.htm` oder ähnlich) vorhanden ist. Die Option `FollowSymLinks` erlaubt Ihnen, innerhalb des `Directory`-Pfad es auch symbolische Links zu nutzen. Alle Optionen können Sie durch ein vorangestelltes Minuszeichen (etwa `-FollowSymLinks`) explizit abschalten.

8.1.3 Debian/Ubuntu: Virtuelle Hosts aktivieren

Wie Sie am Verzeichnisnamen vermutlich schon festgestellt haben, ist der virtuelle Host auf Debian und Ubuntu jetzt konfiguriert, aber noch nicht aktiv. Um ihn zu aktivieren, setzen Sie im Verzeichnis `sites-enabled` einen Symlink auf Ihre Konfigurationsdatei, die ja unter `sites-available` liegt, und lassen die Serverkonfiguration mit `root`-Rechten neu einlesen:


```
$ ln -s /etc/apache2/sites-available/example.com.conf \
/etc/apache2/sites-enabled/example.com.conf
```

Listing 8.2 Virtuellen Host aktivieren

Das Gleiche geht auch kürzer mit dem `a2ensite`-Kommando:

```
$ a2ensite example.com.conf
Enabling site example.com.
To activate the new configuration, you need to run:
    systemctl reload apache2
```

Listing 8.3 Virtuellen Host mit »a2ensite« aktivieren

Wollen Sie Ihren virtuellen Host wieder deaktivieren, benutzen Sie das Kommando `a2dissite`, oder löschen Sie den Symlink, gefolgt von einem Server-Reload. Unter CentOS oder openSUSE Leap ist dieser Schritt nicht notwendig, da die Konfiguration des virtuellen Hosts direkt eingebunden wird.

Default

Die Verarbeitung der virtuellen Hosts geschieht alphabetisch. Daher lautet der Name der Konfigurationsdatei auf Debian- und Ubuntu-Systemen auch `000-default.conf`. Somit wird sichergestellt, dass diese Datei stets zuerst geladen wird. Falls der Apache bei einer Anfrage keine passende Konfiguration findet, wird der Default verwendet. Dies ist entweder die zuerst gefundene Konfiguration oder die, die keine `ServerName`- oder `ServerAlias`-Direktive enthält, wodurch sie universal wird.



Kein Zugriff? Firewallfreigabe einrichten!

Falls Sie trotz der obigen Konfiguration stets eine Fehlermeldung im Browser erhalten, müssen Sie unter Umständen zusätzliche Regeln in der lokalen Firewall einrichten:

```
### openSUSE Leap und CentOS:
server:~ # firewall-cmd --add-service=http
server:~ # firewall-cmd --add-service=https
server:~ # firewall-cmd --permanent --add-service=http
server:~ # firewall-cmd --permanent --add-service=https
```



8.1.4 HTTPS konfigurieren

Um Webseiten verschlüsselt auszuliefern, benötigt Ihr Webserver zunächst ein Serverzertifikat. Zu Testzwecken oder für den Eigenbedarf können Sie sich ein solches Zertifikat schnell und einfach selbst erstellen. Sie müssen nur damit leben, dass Ihr Browser beim Ansurfen der HTTPS-geschützten Webseite eine Warnung ausgibt, weil das selbst erstellte Zertifikat

nicht von einer vertrauenswürdigen Instanz signiert wurde. Da unbedarfte Nutzer von dieser Warnung leicht abgeschreckt werden, sollten Sie für Ihre Produktivsysteme auf signierte Zertifikate kommerzieller Anbieter zurückgreifen. Für die Beispiele in diesem Kapitel genügt natürlich ein selbst erstelltes Zertifikat.



Mehr zu Zertifikaten

Alle Details zu Zertifikaten haben wir für Sie in Kapitel 31, »Verschlüsselung und Zertifikate«, zusammengetragen. Schauen Sie sich dort insbesondere Abschnitt 31.4, »Einmal mit allem und kostenlos bitte: »Let's Encrypt«, an. Dort zeigen wir Ihnen, wie Sie kostenlose SSL-Zertifikate erstellen können, die von allen Browsern akzeptiert werden.

So erstellen Sie den Pfad und ein Serverzertifikat:

```
$ mkdir /etc/apache2/ssl      ### CentOS: /etc/httpd/
$ cd /etc/apache2/ssl
$ openssl req -new -x509 -keyout myserver.pem -out myserver.pem -days 365 -nodes
```

Listing 8.4 Verzeichnis für das SSL-Zertifikat anlegen

OpenSSL stellt Ihnen eine Reihe von Fragen zu Ihrem Standort und Kontaktadresse, die Sie nach Belieben beantworten können oder auch nicht – bei einem selbst signierten Testzertifikat sind diese Informationen nicht wesentlich.



Passen Sie aber auf, wenn die Frage nach dem *Common Name* erscheint! Hier müssen Sie unbedingt den exakten Namen eingeben, unter dem Ihr HTTPS-Webangebot später erreichbar sein soll. Wollen Sie Ihre Webseite also unter `https://www.example.com` aufrufen können, so müssen Sie auf die Frage nach dem *Common Name* mit `www.example.com` antworten.

Debian/Ubuntu

Zur Aktivierung von SSL müssen die Statements aus Listing 8.5 in der Datei `ports.conf` vorhanden sein – falls die Zeilen mit Kommentarzeichen (#) beginnen, entfernen Sie diese und verlassen die `ports.conf`.

```
<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>
```

Listing 8.5 Das »Listen 443«-Statement in »ports.conf«

Führen Sie nun die beiden folgenden Kommandos aus, um das SSL-Modul zu aktivieren und Apache neu zu starten:

```
$ a2enmod ssl
$ systemctl restart apache2
```

Listing 8.6 Debian/Ubuntu: SSL-Modul aktivieren

openSUSE Leap

Ähnlich wie unter Debian und Ubuntu muss auch unter openSUSE Leap die Listen-Direktive gesetzt werden. Hier verbirgt sie sich aber in der Datei */etc/apache2/listen.conf*, wie Listing 8.7 zeigt:

```
<IfDefine SSL>
  <IfDefine !NOSSL>
    <IfModule mod_ssl.c>
      Listen 443
    </IfModule>
  </IfDefine>
</IfDefine>
```

Listing 8.7 Das »Listen 443«-Statement in »listen.conf«

Auch hier müssen Sie sicherstellen, dass die Zeilen nicht auskommentiert sind. Ebenso müssen Sie, wie es in Listing 8.8 dargestellt ist, mit den folgenden Kommandos das SSL-Modul aktivieren und den Apache neu starten:

```
$ a2enmod ssl
$ systemctl restart apache2
```

Listing 8.8 openSUSE: SSL-Modul aktivieren

CentOS

Per Default verfügt der Apache unter CentOS nicht über SSL-Unterstützung. Um diese zu erhalten, müssen Sie das Paket *mod_ssl* installieren (siehe Listing 8.9). Bei der Installation wird die notwendige Konfiguration direkt mitgeliefert. Sie finden diese in der Datei */etc/httpd/conf.d/ssl.conf*. Dort ist direkt auch ein virtueller Host eingerichtet, der als Default-Webseite dient.

```
[root@centos ~]# dnf install mod_ssl
```

Listing 8.9 Installation des SSL-Moduls auf CentOS

HTTPS-Konfiguration eines virtuellen Hosts

Apache ist jetzt grundsätzlich in der Lage, HTTPS-Anfragen zu bedienen. Sie benötigen also nur noch einen virtuellen Host, der sie auch beantwortet.

Legen Sie im entsprechenden Verzeichnis für die Konfigurationsdateien die Datei `ssl_example.com.conf` mit dem Inhalt aus Listing 8.10 an:

```
<VirtualHost *:443>
    ServerName www.example.com
    ServerAdmin webmaster@example.com
    DocumentRoot /var/www/example.com/
    ### openSUSE Leap: /srv/www/vhosts/example.com
    ErrorLog /var/log/apache2/error.log
    CustomLog /var/log/apache2/example.com.log combined
    ### CentOS: /var/log/httpd/

    SSLEngine On
    SSLCertificateFile /etc/apache2/ssl/myserver.pem

    <Directory /var/www/example.com/>
    ### openSUSE Leap: /srv/www/vhosts/example.com
        Options Indexes FollowSymLinks
        ### openSUSE Leap:
        # AllowOverride None
        # <IfModule !mod_access_compat.c>
        #     Require all granted
        # </IfModule>
        # <IfModule mod_access_compat.c>
        #     Order allow,deny
        #     Allow from all
        # </IfModule>
        ###
    </Directory>
</VirtualHost>
```

Listing 8.10 Konfigurationsdatei für einen virtuellen SSL-Host



Mehrere virtuelle SSL-Hosts mit einer IP-Adresse

Im Gegensatz zu »normalen« virtuellen Hosts – also solchen ohne SSL – können Sie nicht beliebig viele virtuelle SSL-Hosts auf einer IP-Adresse konfigurieren. Sie benötigen für jeden SSL-Host eine eigene IP-Adresse. Um dieser Verschwendung von IP-Adressen entgegenzutreten, wurde eine Erweiterung geschaffen: *Server Name Indication*.

Beachten Sie dabei, dass auch bei dieser Konfiguration die Pfade an Ihre Distribution angepasst werden müssen. Auf openSUSE-Leap-Systemen müssen Sie zusätzlich in der Datei `/etc/sysconfig/apache2` die nachstehende Direktive um den Wert `SSL` erweitern:

```
APACHE_SERVER_FLAGS="SSL"
```

Listing 8.11 openSUSE Leap: Anpassung für SSL in »/etc/sysconfig/apache2«

Nach dem obligatorischen `systemctl restart apache2` (bzw. `httpd`) können Sie Ihre HTTPS-Seite jetzt im Browser aufrufen. Da Ihr Server dem Browser ein selbst signiertes Zertifikat präsentiert, werden Sie zunächst mit einer Warnmeldung wie in Abbildung 8.1 konfrontiert.

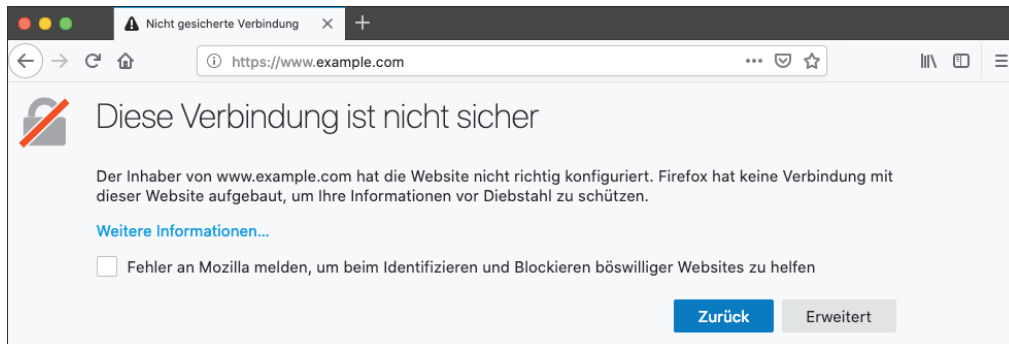


Abbildung 8.1 Der Browser erhält ein selbst signiertes Zertifikat.

Der Browser verlangt nun (einmalig) von Ihnen, dass Sie das Zertifikat vom Server laden, anschauen und dann ausdrücklich akzeptieren (siehe Abbildung 8.2).

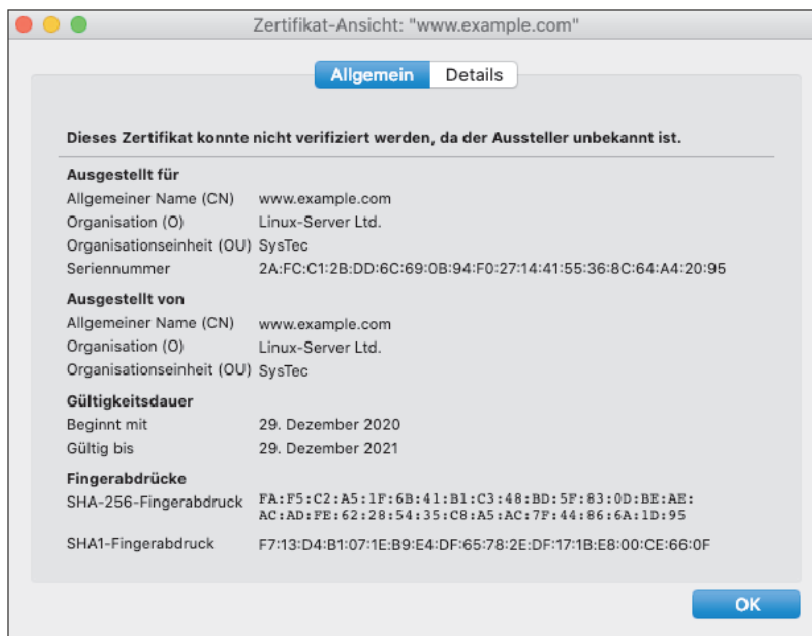


Abbildung 8.2 Das selbst signierte Zertifikat

Ihr Browser merkt sich, dass Sie dem Zertifikat vertrauen, und wird Sie bei einem erneuten Besuch der Webseite nicht mehr fragen. Damit haben Sie die Konfiguration Ihres virtuellen SSL-Hosts abgeschlossen.

8.1.5 Apache-Server mit ModSecurity schützen

ModSecurity nennt sich selbst »Web Application Firewall (WAF)«. Es ist ein Apache-Modul, das den HTTP-Datenverkehr mithilfe eines Regelwerks auf Angriffsmuster prüft. Der Unterschied zu klassischen Firewalls besteht in der OSI-Schicht, auf der der Filter arbeitet. Ein Paketfilter arbeitet auf der Transportebene, also der OSI-Schicht 3 oder 4. Er muss seine Entscheidungen anhand von Quelle, Ziel und Status des Pakets treffen. Die Web Application Firewall werkelt dagegen auf der gleichen Ebene wie der Webserver selbst (auf der Applikationsebene 7) und kann so den Inhalt der Pakete analysieren.

Die Motivation hinter dieser Vorgehensweise ist eine Verschiebung der Angriffsvektoren. Früher zielten Angreifer darauf ab, Sicherheitslücken in der Serversoftware auszunutzen. Durch den jahrelangen Reifeprozess sind klaffende Sicherheitslücken jedoch selten geworden, und auftretende Sicherheitsprobleme werden schnell durch Updates behoben. Anders sieht es bei den Webangeboten selbst aus. Viele werden nicht mit einem Fokus auf der Sicherheit entwickelt und enthalten Schnittstellen zu anderen Diensten wie SQL-Servern oder Verzeichnisdiensten. Angreifer versuchen heute nicht mehr, den Server selbst anzugreifen, sondern wollen über geschickt formulierte HTTP-Anfragen Daten stehlen oder eigenen Code ausführen lassen.

Genau diese HTTP-Anfragen versucht *ModSecurity* mithilfe seines Regelwerks zu erkennen und herauszufiltern. *ModSecurity* wird derzeit von der Firma *Trustwave SpiderLabs* weiterentwickelt, die neben der Open-Source-Version auch einen kommerziellen Zweig anbietet. Die Software kann direkt auf dem Server eingebunden werden oder als Proxy mehreren Servern vorgeschaltet werden.

Installation

Die meisten Distributionen stellen *ModSecurity* als Paket zur Verfügung. Debian war lange Zeit eine Ausnahme, aber das hat sich inzwischen geändert. Debian 6.0 »Squeeze« war die erste stabile Debian-Version, in der *ModSecurity* enthalten ist.

Die Installation erfolgt über die Paketquellen:

- ▶ **Debian:** `apt install libapache2-mod-security2`
- ▶ **Ubuntu:** `apt install libapache2-mod-security2`
- ▶ **CentOS:** `dnf install mod_security`
- ▶ **openSUSE Leap:** `zypper install apache2-mod_security2`

Debian/Ubuntu

Neben dem eigentlichen Modul lebt ModSecurity vom Regelwerk. Dieses laden wir zunächst in der aktuellen Version vom GitHub-Repository des Herstellers herunter. Anschließend müssen Sie das Regelwerk noch konfigurieren. Dafür müssen lediglich die Beispielkonfigurationsdateien an ihren Bestimmungsort gebracht werden (siehe Listing 8.12).

```
### Herunterladen
$ rm -rf /usr/share/modsecurity-crs
$ git clone https://github.com/coreruleset/coreruleset.git \
  /usr/share/modsecurity-crs
```

```
### Konfiguration & Regeln aktivieren
$ cd /usr/share/modsecurity-crs
$ cp crs-setup.conf.example /etc/modsecurity/crs-setup.conf
```

Listing 8.12 Aktuelles Regelwerk herunterladen und Konfiguration und Regeln aktivieren

Im Paket ist bereits eine umfangreiche Beispielkonfiguration vorhanden. Damit diese auch verwendet wird, muss sie nun einen neuen Namen erhalten:

```
$ cd /etc/modsecurity
$ cp modsecurity.conf-recommended modsecurity.conf
```

Listing 8.13 Kopieren der Beispielkonfiguration »modsecurity.conf«

Per Default werden Regelverstöße lediglich dokumentiert. Damit ModSecurity auch wirklich arbeitet, muss nun die Direktive `SecRuleEngine` auf `On` gesetzt werden:

```
# SecRuleEngine DetectionOnly
SecRuleEngine On
```

Listing 8.14 Anpassung in »/etc/modsecurity/modsecurity.conf«

Zu guter Letzt müssen Sie noch in der Modulkonfiguration die neu heruntergeladenen Regeln hinzufügen. Öffnen Sie dafür die Datei `/etc/apache2/mods-available/security2.conf`, und fügen Sie die in Fettschrift dargestellte Zeile aus Listing 8.15 hinzu:

```
<IfModule security2_module>
    # Default Debian dir for modsecurity's persistent data
    SecDataDir /var/cache/modsecurity

    # Include all the *.conf files in /etc/modsecurity.
    # Keeping your local configuration in that directory
    # will allow for an easy upgrade of THIS file and
    # make your life easier
    IncludeOptional /etc/modsecurity/*.conf
    # Include OWASP ModSecurity CRS rules if installed
```

```

    IncludeOptional /usr/share/modsecurity-crs/rules/*.conf
</IfModule>

```

Listing 8.15 Anpassung in »/etc/apache2/mods-available/security2.conf«

Mit den in Listing 8.16 gezeigten Kommandos aktivieren Sie das Modul und starten den Webserver neu, damit die WAF ihren Dienst auch antreten kann:

```

$ a2enmod security2
$ systemctl restart apache2

```

Listing 8.16 Aktivieren des Moduls und Neustart des Apache

CentOS

Auch auf CentOS-Systemen sollte zunächst das aktuelle Regelwerk geladen werden. Anschließend müssen Sie sowohl die Konfiguration als auch die Regeln aktivieren (siehe Listing 8.17).

```

### Herunterladen
$ cd /etc/httpd/modsecurity.d/
$ git clone https://github.com/coreruleset/coreruleset.git
$ cp coreruleset/rules/* activated_rules/

### Konfigurieren & Aktivieren
$ cd /etc/httpd/modsecurity.d/
$ cp coreruleset/crs-setup.conf.example crs-setup.conf

```

Listing 8.17 Regelwerk herunterladen, konfigurieren und aktivieren unter CentOS

Die übrige Konfiguration ist bereits Bestandteil des Pakets, sodass keine weiteren Arbeiten notwendig sind. Nach dem obligatorischen Neustart des Webserver ist die WAF aktiv. Allerdings könnte auch hier erforderlich sein die Regel 922 zu löschen da gegebenenfalls die ModSecurity-Version zu alt ist, um mit den aktuellen Regeln arbeiten zu können.



Fehler: »Unknown variable: &MULTIPART_PART_HEADERS«

Ab den CRS Versionen 3.2.2 oder 3.3.3 und höher wird mindestens eine ModSecurity Version 2.9.6 oder 3.0.8 benötigt, ansonsten erhalten Sie nachstehenden Fehler wenn Sie den Webserver neu starten:

```

[...]: AH00526: Syntax error on line 43 of /usr/share/modsecurity-crs/rules/\
        REQUEST-922-MULTIPART-ATTACK.conf:
[...]: Error creating rule: Unknown variable: &MULTIPART_PART_HEADERS

```

Um diesen zum Umgehen sollten Sie eine Update durchführen - ist dies nicht möglich können Sie sich behelfen indem Sie die Regel 922 einfach löschen. Allerdings verzichten Sie damit auf einige modernere Schnutzmechanismen.

openSUSE Leap

Nach der Installation müssen die Module so aktiviert werden, wie es Listing 8.18 zeigt:

```
$ a2enmod security2
$ a2enmod unique_id
```

Listing 8.18 Aktivieren der Module für »ModSecurity«

Bei openSUSE Leap wird direkt ein Regelwerk mitinstalliert – das allerdings stark veraltet ist. Daher laden wir trotzdem die aktuellen Regeln herunter und speichern sie an der dafür vorgesehenen Stelle:

```
### backup, download und kopieren
$ cd /usr/share/apache2-mod_security2/
$ mv rules rules.BAK
$ git clone https://github.com/coreruleset/coreruleset.git
$ cp -r coreruleset/rules .

### Regeln aktivieren
$ cd /etc/apache2/mod_security2.d/
for f in /usr/share/apache2-mod_security2/rules/*; do ln -s $f . ; done

### Regeln konfigurieren
$ cd /usr/share/apache2-mod_security2/
$ cp -r coreruleset/crs-setup.conf.example rules/modsecurity_crs_10_setup.conf
```

Listing 8.19 Aktivieren des Regelwerks auf openSUSE Leap

Die übrige Konfiguration ist bereits bei der Paketinstallation vorgenommen worden, sodass keine weiteren Arbeiten notwendig sind. Nach dem obligatorischen Neustart des Apache mit `systemctl restart apache2` ist die WAF aktiv. Auch an dieser Stelle müssen wir auf den aktuellen Fehler aufgrund der unterschiedlichen Versionen hinweisen.



Konfiguration

Das Standardregelwerk, das ModSecurity mitbringt, ist schon recht restriktiv und blockiert oft mehr, als erwünscht ist. Das kann dazu führen, dass plötzlich Teile Ihrer Webanwendung nicht mehr aufrufbar sind, weil ModSecurity die entsprechenden Anfragen für gefährlich hält. Um die Wirksamkeit der Regeln ohne Gefahr für Ihren Blutdruck auszuprobieren, können Sie ModSecurity in einen »Detection«-Modus schalten. Dabei durchlaufen alle Anfragen das Regelwerk, eventuelle Treffer werden aber lediglich ins Logfile geschrieben.

Diesen Modus aktivieren Sie, indem Sie in der Datei `crs-setup.conf` (openSUSE Leap: `mod-security_crs_10_config.conf`) die Zeile `SecRuleEngine On` auskommentieren und wie im folgenden Beispiel ersetzen:

```
#SecRuleEngine On
SecRuleEngine DetectionOnly
```

Listing 8.20 »ModSecurity« in den »DetectionOnly«-Modus schalten

Wenn ModSecurity einen Zugriff ablehnt, finden Sie im Audit-Logfile das Sie unter `/var/log/apache2/modsec_audit.log` (CentOS: `/var/log/httpd/modsec_audit.log`) finden und in den jeweiligen Error-Logs der entsprechenden Seite, Zeilen wie diese:

```
Message: Warning. Matched phrase "etc/passwd" at ARGS:file.
[file "/usr/share/modsecurity-crs/rules/REQUEST-930-APPLICATION-ATTACK-LFI.conf"]
[line "116"]
[id "930120"]
[msg "OS File Access Attempt"]
[data "Matched Data: etc/passwd found within ARGS:a: etc/passwd"]
[severity "CRITICAL"]
[ver "OWASP_CRS/4.0.0-rc1"]
[tag "application-multi"]
[tag "language-multi"]
[tag "platform-multi"]
[tag "attack-lfi"]
[tag "paranoia-level/1"]
[tag "OWASP_CRS"]
[tag "capec/1000/255/153/126"]
[tag "PCI/6.5.4"]
```

Listing 8.21 »ModSecurity« lehnt einen Zugriff ab.

Hier bekommen Sie genaue Informationen darüber, welche Regel wirksam wurde, in welcher Konfigurationsdatei und sogar in welcher Zeile. Jede Regel hat eine eindeutige Kennnummer; im Beispiel aus Listing 8.21 ist es `[id "930120"]`. Wenn Sie der Meinung sind, dass ModSecurity übereifrig war und den Zugriff zu Unrecht unterbunden hat, können Sie die Regel deaktivieren.



Erstellen Sie eigene Konfigurationsdateien!

Es ist nicht sinnvoll, die Regeln des Kernregelwerks direkt zu editieren. Das funktioniert zwar zunächst, rächt sich aber beim nächsten Update des Regelwerks. Benutzen Sie eine oder mehrere eigene Konfigurationsdateien, um Regeln hinzuzufügen oder zu deaktivieren.

Wechseln Sie in das Verzeichnis, in dem Ihr ModSecurity-Regelwerk residiert, und aktivieren Sie die für Ausnahmen vorgesehenen Dateien `REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example` und `RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.example` um, in-

dem Sie die Endung *.example* entfernen. Möchten Sie dann beispielsweise die Regel mit der ID 930120 deaktivieren, ergänzen Sie in *RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf* den folgenden Inhalt:

```
SecRuleRemoveById 930120
```

Listing 8.22 Eine Regel deaktivieren

Eine Regel kommt selten allein!

Meistens ist es nicht mit der Ausnahme nur einer Regel getan. Für das Beispiel aus Listing 8.21 genügt es nicht, nur die Regel mit der ID 930120 zu deaktivieren, tatsächlich müssten Sie noch die Regeln mit den IDs 932160, 949110 und 980170 ebenfalls deaktivieren, damit der Zugriff nicht mehr unterbunden wird – was wir aber natürlich nicht wollen! Die Datei */etc/passwd* sollte schließlich niemals öffentlich zugänglich sein.



8.1.6 Tuning und Monitoring

Leistungsoptimierung fängt bei der Hardware an. Dabei ist die reine Rechenleistung des Servers eher selten ein limitierender Faktor, wenn Sie nicht gerade hochkomplexe dynamische Webangebote betreiben. Viel eher kommt es auf den Speicher an. Kaum eine Tuning-Maßnahme ist so wirksam wie das Aufstocken des verfügbaren RAMs. Die Leistung jedes Webserver bricht schlagartig und dramatisch ein, wenn das System in den Swap-Betrieb schlittert.

Aber auch dann, wenn Sie Ihrem Server ausreichend Speicher spendiert haben, können Sie durch einige einfache Handgriffe noch etwas mehr Leistung herauskitzeln. Außerdem bringt Apache bereits ein Bordmittel mit, das Ihnen einen Blick unter die Motorhaube erlaubt.

Multi-Processing-Module

Zur Verarbeitung der eingehenden HTTP-Anfragen greift Apache auf eines von mehreren, teils plattformabhängigen *Multi-Processing-Modulen* (MPMs) zurück. Drei davon sind unter Linux besonders relevant:

- ▶ **Prefork-MPM:** Mit diesem Modul startet Apache eine Reihe von Child-Prozessen mit jeweils nur einem einzigen Thread. Es eignet sich gut für ältere Software, die Probleme in Multithreading-Umgebungen hat.
- ▶ **Worker-MPM:** Dieses MPM ist Multithreading-fähig, benötigt weniger RAM als das Prefork-MPM und eignet sich generell besser für stark belastete Server.
- ▶ **Event-MPM:** Das Event-MPM ist eine Variante des Worker-MPMs, die etwas geschickter mit Keep-Alive-Verbindungen umgeht. Für diese stellt es einen eigenen Thread zur Verfügung, sodass der ursprüngliche Thread schneller für neue Verbindungen frei wird.

Die »MaxRequestWorkers«-Direktive

In den Konfigurationsdateien finden Sie unter anderem die `MaxRequestWorkers`-Direktive. Sie gibt an, wie viele Clientverbindungen maximal bedient werden können (beim Prefork-MPM) oder wie viele Threads für Clientverbindungen zur Verfügung stehen (beim Worker- und Event-MPM).



Per Default sind dies 256, manche Distributionen arbeiten aber mit anderen Voreinstellungen. Unter Debian ist der `MaxRequestWorkers`-Wert auf 150 eingestellt.

Sollte der voreingestellte Wert zu niedrig sein, können nicht alle Clients bedient werden. Sie erkennen das Problem an Meldungen im Error-Log, die etwa so aussehen wie in Listing 8.23:

```
[Wed Dec 30 13:15:24.760818 2022] [mpm_prefork:error] [pid 1648] AH00161: server \
reached MaxRequestWorkers setting, consider raising the MaxRequestWorkers setting
```

Listing 8.23 Fehlermeldung wegen zu niedriger »MaxRequestWorkers«-Einstellung

Erhöhen Sie in diesem Fall den Wert, aber behalten Sie, etwa mit `top`, Ihren RAM-Verbrauch im Auge. Ein höherer `MaxRequestWorkers`-Wert geht unweigerlich mit erhöhten Speicheranforderungen einher.

DNS-Anfragen

Apache protokolliert alle Zugriffe in einem Logfile. Damit Sie leicht erkennen können, welche Clients auf Ihren Webserver zugegriffen haben, versucht Apache immer, durch DNS-Anfragen den Namen des Clients zu ermitteln.

Diese Anfragen bestehen in der Regel zwar nur aus recht kleinen UDP-Paketen, aber wenn Sie einen stark frequentierten Webserver betreiben, können die *Reverse Lookups* trotzdem bremsend wirken. Fügen Sie der Konfigurationsdatei Ihrer (virtuellen) Hosts die in Listing 8.24 gezeigte Direktive hinzu:

```
[...]
HostnameLookups Off
[...]
```

Listing 8.24 DNS-Anfragen deaktivieren

Auf die Information, welche Clients Ihre Webseite besucht haben, müssen Sie dennoch nicht verzichten.

Der »htaccess«-Mechanismus

Der `htaccess`-Mechanismus erlaubt es Ihnen, bestimmte Konfigurationsoptionen pro Verzeichnis ein- oder auszuschalten, und trägt damit wesentlich zur Flexibilität Ihres Servers bei. Falls Sie diese Flexibilität nicht benötigen, sollten Sie den Mechanismus unbedingt abschalten. Sie ersparen Ihrem Apache damit unzählige unnötige Lesevorgänge und Fall-

entscheidungen. Fügen Sie die Zeilen aus Listing 8.25 der Konfigurationsdatei Ihres (virtuellen) Hosts hinzu, um den *htaccess*-Mechanismus zu deaktivieren:

```
[...]
<Directory />
AllowOverride none
</Directory>
[...]
```

Listing 8.25 »htaccess« deaktivieren

Anzeigen des Serverstatus

Apaches Statusmodul *mod_status* erlaubt es Ihnen, den Apache-Prozessen zu jedem Zeitpunkt auf die Finger zu schauen. Sie aktivieren das Modul auf Debian-, Ubuntu- und openSUSE-Systemen mit dem bereits bekannten Programm *a2enmod*. Wie gewohnt wird dem Programm nur das entsprechende Modul als Parameter übergeben: *a2enmod status*.

Nach einem Neustart des Apache können Sie die Statuswebseite unter der Adresse *www.example.com/server-status* betrachten – theoretisch. In der Praxis darf per Default nur *localhost* auf die Statusseite zugreifen. Andere Rechner müssen Sie explizit freigeben. Das können Sie in der Konfigurationsdatei */etc/apache2/mods-available/status.conf* (Debian/Ubuntu) oder */etc/apache2/mod_status.conf* (openSUSE Leap) tun. Die Datei enthält den Konfigurationsblock aus Listing 8.26:

```
# Allow server status reports generated by mod_status,
# with the URL of http://servername/server-status
# Uncomment and change the "192.0.2.0/24" to allow access from other hosts.

<Location /server-status>
    SetHandler server-status
    Require local
    #Require ip 192.0.2.0/24
</Location>
```

Listing 8.26 Auszug der Konfigurationsdatei des Statusmoduls

Entfernen Sie das Kommentarzeichen vor der in Fettschrift dargestellten Zeile, und passen Sie das Netzwerk an Ihre Umgebung an, oder tragen Sie nur die IP-Adresse Ihres Clients ein.

Neben IP-Adressen sind auch Hostnamen und die Angabe von Subnetzen erlaubt. Alternativ können Sie auch eine oder mehrere zusätzliche *Require*-Zeilen einfügen:

```
Require ip 10.20.30.40
Require host client01.lan.example.com
Require ip 172.31.0.0/16
```

Listing 8.27 »Require«-Statements im Statusmodul

Bei CentOS müssen Sie die Konfigurationsdatei zunächst selbst anlegen. Speichern Sie den Inhalt aus Listing 8.28 in die Datei `/etc/httpd/conf.d/server-status.conf`:

```
<IfModule mod_status.c>
    <Location /server-status>
        SetHandler server-status
        Require local
        Require ip 192.168.1.0/24    ### <-- Anpassung an Ihr Netzwerk!
    </Location>

    ExtendedStatus On
    <IfModule mod_proxy.c>
        # Show Proxy LoadBalancer status in mod_status
        ProxyStatus On
    </IfModule>
</IfModule>
```

Listing 8.28 Konfiguration des Statusmoduls für CentOS

Nach einem Apache-Reload können Sie die Statusseite aufrufen. Sie sieht etwa so aus, wie Listing 8.29 zeigt:

```
Server Version: Apache/2.4.52 (Ubuntu) OpenSSL/3.0.2
Server MPM: prefork
Server Built: 2022-09-30T04:09:50
[...]
Current Time: Wednesday, 28-Dec-2022 16:06:56 UTC
Restart Time: Wednesday, 28-Dec-2022 15:47:10 UTC
Parent Server Config. Generation: 2
Parent Server MPM Generation: 1
Server uptime: 13 minutes 43 seconds
Server load: 0.14 0.03 0.01
Total accesses: 2835168 - Total Traffic: 414.3 GB - Total Duration: 21
CPU Usage: u187.68 s76.8281 cu0 cs0 - .637% CPU load
68.3 requests/sec - 10.2 MB/second - 153.2 kB/request - 2.25 ms/request
305 requests currently being processed, 143 idle workers
[...]
_W_KKWWWWW_WW_WCWKWWWWCWKKKW_WWWWWW_WW_WWWWK_WWWWK_WWK_WWWKW_WWKW
_____KC_____KWK_W_W_W_KKKKK_KCK_WW_CKWWC_KK_KWC_____
KKW_WKW_W_CK_K_C_WC_____W_W_K_____KW_W_K_KKW_W_C_WK_____
WK_WKW_C_KKKKWWW_W_W_W_KK_KW_WWWC_KW_W_WK_WWWKWW_W_KW_WWW
WWWK_WKCWKWKKKWWWCWKWK_W_CKWWCW_KW_W_K_WKWWWKCC_W_KWWWKWWK
[...]
```

Listing 8.29 Apache-Serverstatus (Auszug)

Der zunächst etwas unsortiert anmutende Zeichenblock zeigt an, womit jeder Apache-Prozess im Moment beschäftigt ist. Tabelle 8.1 listet die Bedeutung der Zeichen auf.

Zeichen	Bedeutung
_	wartet auf Verbindung
S	Prozess startet
R	Anfrage des Clients wird empfangen.
W	Antwort wird gesendet.
K	Keep-Alive – die Verbindungen bleiben für weitere Anfragen offen.
D	Nameserver-Lookup
C	Verbindung wird geschlossen.
L	Eintrag wird ins Logfile geschrieben.
G	Ein Prozess wird kontrolliert beendet.
I	<i>Idle Cleanup</i> – ein Prozess erhielt keine Verbindungen und wird aufgeräumt.
.	Verbindungs-Slot ohne laufenden Prozess

Tabelle 8.1 Statuskürzel in der Serverstatusanzeige

8.2 nginx

nginx ist ein schlanker und schneller Webserver, der dem Apache mächtig Konkurrenz macht. Insbesondere das Ausliefern von statischem Inhalt erledigt *nginx* (man spricht den Namen als *Engine X* aus) extrem schnell bei gleichzeitig sehr geringem RAM-Bedarf. Ebenso zeichnet *nginx* sich dadurch aus, dass neue Technologien sehr schnell unterstützt werden.

8.2.1 Installation

nginx gehört zum Standardumfang jeder größeren Distribution, daher kann die Installation einfach über die Paketquellen erfolgen:

- ▶ **Debian:** `apt install nginx`
- ▶ **Ubuntu:** `apt install nginx`
- ▶ **CentOS:** `dnf install nginx`
- ▶ **openSUSE Leap:** `zypper install nginx`

8.2.2 Grundlegende Konfiguration

Unter `/etc/nginx` finden Sie die Konfigurationsdateien von nginx. In der Datei `nginx.conf` finden Sie grundsätzliche Einstellungen, HTTP-Optionen und Möglichkeiten, um auf die Performance Einfluss zu nehmen. Ähnlich wie beim Apache ist die eigentliche Konfiguration der (virtuellen) Hosts in separate Dateien ausgegliedert. Ihre Konfiguration wird im nächsten Abschnitt behandelt, hier schauen wir uns zunächst die Basiskonfiguration in der `/etc/nginx/nginx.conf` an. Sie finden hier bereits eine funktionierende Konfiguration vor, die aber in der Regel noch Optimierungspotenzial hat. Folgende Einstellungen können Sie hinzufügen oder, wenn sie schon vorhanden sind, an Ihre Bedürfnisse anpassen:

- ▶ **worker_processes auto;**
nginx fragt die Anzahl der (virtuellen) CPUs selbstständig ab und skaliert entsprechend. Das »harte« Einstellen der Worker-Prozesse auf einen festen Wert ist daher nicht mehr notwendig.
- ▶ **worker_rlimit_nofile 300000;**
Beim Apache sind die systemweiten Limits für die Anzahl gleichzeitig offener Files unkritisch, weil sie pro (Sub-)Prozess gelten und Apache für jeden Request einen neuen Thread öffnet. Bei nginx ist das Limit aber unter Umständen ein Problem und sollte großzügig nach oben korrigiert werden. Dieser Wert ist standardmäßig nicht gesetzt.
- ▶ **multi_accept on**
Da nginx schon per Default *non-blocking* arbeitet, sollte man das Abarbeiten von Requests nicht künstlich serialisieren. Dieser Wert ist standardmäßig nicht gesetzt und steht auf `off`.
- ▶ **worker_connections 2048;**
Per Default verarbeitet nginx maximal 512 gleichzeitige Requests pro Worker-Prozess. Es spricht nichts dagegen, diesen Wert auf leistungsfähiger Hardware anzuheben.
- ▶ **sendfile on;**
tcp_nopush on;
Diese Einstellungen werden im `http`-Abschnitt der `/etc/nginx/nginx.conf` vorgenommen. `Sendfile()` ist der *System Call*, der Dateien von der Platte in ein TCP-Paket bringt. Es arbeitet dabei deutlich effizienter als ältere Verfahren. `tcp_nopush` sorgt dafür, dass Header-Daten in einem einzigen Paket (statt mehreren) geschickt werden.

8.2.3 Virtuelle Hosts

Für die folgenden Beispiele wollen wir zunächst einen Server konfigurieren, der auf die Namen `www.example.com` und `example.com` hört. Die Daten, die er ausliefert, liegen im Verzeichnis `/var/www/example.com`. Danach konfigurieren wir die Subdomain `sub.example.com`, die Daten aus `/var/www/sub-example.com` ausliefert.



Verzeichnisse erstellen unter CentOS und openSUSE Leap

Die Verzeichnisse *sites-available* und *sites-enabled* sind standardmäßig unter CentOS und openSUSE Leap nicht vorhanden. Daher müssen Sie diese zunächst anlegen.

Legen Sie zunächst die Konfigurationsdatei `/etc/nginx/sites-available/example.com` an. Sie enthält den sogenannten `server`-Konfigurationsblock. In diesem Konfigurationsblock definieren Sie mindestens folgende Einstellungen:

- ▶ **server_name** und **listen**
die IP-Adresse und den Port, auf dem Ihr Server auf eingehende Verbindungen wartet
- ▶ **root**
Diese Einstellung hat nichts mit dem Benutzer `root` zu tun, sondern entspricht dem `DocumentRoot` beim Apache. Hier wird definiert, wo die auszuliefernden Dateien liegen.
- ▶ **location**
Hier legen Sie die Namen der Index-Dateien fest.

Eine gültige Konfigurationsdatei für eine Serverkonfiguration kann also etwa so aussehen wie in Listing 8.30 dargestellt.

```
server{
    listen      80;
    server_name example.com www.example.com;
    root        /var/www/example.com;
    location / {
        index   index.html index.htm;
    }
}
```

Listing 8.30 Eine einfache Server-Konfigurationsdatei

Wenn Sie nun zusätzlich die Subdomain `sub.example.com` einrichten wollen, haben Sie die Wahl: Sie können entweder eine weitere Konfigurationsdatei unter `/etc/nginx/sites-available` anlegen, oder Sie schreiben einen zweiten `server`-Konfigurationsblock in die bereits bestehende Datei, die Sie gerade angelegt haben.

Die neue Konfiguration können Sie einfach unter dem bereits bestehenden Teil anhängen, wie in Listing 8.31 gezeigt:

```
server{
    listen      80;
    server_name example.com www.example.com;
    root        /var/www/example.com;
```

```
    location / {
        index    index.html index.htm;
    }
}

server{
    listen      80;
    server_name sub.example.com www.sub.example.com;
    root        /var/www/sub-example.com;

    location / {
        index    index.html index.htm;
    }
}
```

Listing 8.31 Zwei Serverkonfigurationen (Domain und Subdomain) in einer Datei

Virtuellen Host aktivieren und deaktivieren

Damit die Konfiguration wirksam wird, müssen Sie im Verzeichnis */etc/nginx/sites-enabled* einen symbolischen Link auf die Konfigurationsdatei im Verzeichnis */etc/nginx/sites-available* anlegen. Das Kommando dazu sehen Sie in Listing 8.32:

```
$ ln -s /etc/nginx/sites-available/myserver /etc/nginx/sites-enabled/myserver
```

Listing 8.32 Symbolischen Link anlegen

Unter CentOS und openSUSE Leap müssen Sie zusätzlich noch das neu angelegte Verzeichnis *sites-enabled* in der Hauptkonfiguration (*nginx.conf*) hinzufügen. Ergänzen Sie dafür innerhalb des `http`-Blocks die Zeilen aus Listing 8.33:

```
http {
    [...]
    include sites-enabled/*;
    [...]
}
```

Listing 8.33 Inkludieren der ausgelagerten virtuellen Hostdateien



CentOS-SELinux-Anpassung!

Da wir die Standardpfade verändern, muss auf CentOS-Systemen SELinux an die neuen Gegebenheiten gewöhnt werden. Führen Sie dafür den nachstehenden Befehl aus: `chcon -Rt httpd_sys_content_t /var/www/`. Ansonsten erhalten Sie beim Aufruf stets den irreführenden Fehler 403.

Danach muss nginx mit `systemctl restart nginx` einmal neu gestartet werden.

CentOS/openSUSE: Kein Zugriff? Firewallfreigabe einrichten!

Falls Sie trotz der obigen Konfiguration stets eine Fehlermeldung im Browser erhalten, müssen Sie unter Umständen zusätzliche Regeln in der lokalen Firewall einrichten:

```
### openSUSE Leap und CentOS:
server:~ # firewall-cmd --add-service=http
server:~ # firewall-cmd --add-service=https
server:~ # firewall-cmd --permanent --add-service=http
server:~ # firewall-cmd --permanent --add-service=https
```



8.2.4 HTTPS mit nginx

Einen nginx-Server HTTPS-fähig zu machen, erforderte in frühen Versionen noch einen erheblichen Aufwand, oft sogar das Neukompilieren des Servers. Diese Zeiten sind zum Glück vorbei. Inzwischen genügt es, einen privaten Schlüssel und ein Zertifikat zu generieren und dem Server den Pfad dorthin mitzuteilen. Am besten legen Sie unterhalb von `/etc/nginx` ein Verzeichnis namens `ssl` an, in dem Sie Schlüssel und Zertifikate aufbewahren. Führen Sie die `openssl`-Kommandos aus Listing 8.34 aus, um die benötigten Dateien für unsere Beispiel-Domain `example.com` zu generieren:

```
# Verzeichnis erstellen und hineinwechseln
mkdir -p /etc/nginx/ssl && cd $_

# privaten Schlüssel und die Signaturanforderung (CSR) erzeugen:
openssl genrsa -out example.com.key 2048
openssl req -new -key example.com.key -out example.com.csr

# Letzter Schritt: signieren
openssl x509 -req -days 365 -in example.com.csr \
  -signkey example.com.key -out example.com.crt
```

Listing 8.34 Schlüssel und Zertifikat für die Domain »example.com« erzeugen

Zum Schluss teilen Sie nginx im `server`-Konfigurationsblock mit, dass er SSL benutzen soll und wo er den Schlüssel und das Zertifikat findet (siehe Listing 8.35). Nach einem Neustart ist der HTTPS-Server betriebsbereit.

```
server {
    listen          443 ssl;
    ssl_certificate /etc/nginx/ssl/example.com.crt;
    ssl_certificate_key /etc/nginx/ssl/example.com.key;
```

```
server_name      example.com www.example.com;
root             /var/www/example.com;

location / {
    index        index.html index.htm;
}
}
```

Listing 8.35 SSL-Unterstützung für die Domain »example.com« aktivieren

8.3 PHP

Sobald auf Webseiten Inhalte dynamische serverseitig berechnet werden, ist die Chance relativ hoch, dass dies mittels PHP geschieht. PHP ist ein rekursives Akronym und Backronym für »PHP: Hypertext Preprocessor«. Die Programmiersprache wurde bereits 1995 veröffentlicht und gehört somit zur Ursuppe des Internets. In diesem Kapitel zeigen wir Ihnen wie Sie Ihrem Webserver beibringen PHP-Seiten zu interpretieren.

PHP ist in Modulen organisiert, wenn eine von Ihnen eingesetzte Funktion als *unbekannt* tituiert wird, dann fehlt Ihrem System voraussichtlich das entsprechende Modul. Suchen Sie am besten in den Paketquellen danach und installieren Sie es bei Bedarf nach. Die Module folgen dabei dem Namensschema `php-<FUNKTION>`.

8.3.1 Installation

Je nach eingesetztem Webserver unterscheidet sich die Installation von PHP erheblich. Beim Apache wird zum Beispiel (wie für fast alles) ein eigenes Apache-Modul angeboten, so dass die Installation schnell von statten geht. Beim nginx hingegen müssen Sie etwas mehr Aufwand betreiben und den PHP-Dienst separat installieren und in die nginx-Konfiguration einbinden. Beide Varianten stellen wir Ihnen nun im Detail vor.

Apache

Wie bereits erwähnt erfolgt die Einbindung von PHP in den Apache mittels einen Moduls – außer bei Centos, dazu später mehr. Dafür installieren Sie zunächst die jeweiligen Pakete Ihrer Distributionn:

- ▶ **Debian:** `apt install php libapache2-mod-php`
- ▶ **Ubuntu:** `apt install php libapache2-mod-php`
- ▶ **CentOS:** `dnf install php php-fpm`
- ▶ **openSUSE Leap:** `zypper install php apache2-mod_php8`

Bei Debian- und Ubuntu-Systemen sind Sie bereits nach diesem Schritt fertig. Bei openSUSE-Systemen müssen Sie nun lediglich noch das Apache-Modul mittels `a2enmod php` aktivieren und den Apache einmal neustarten.

Bei CentOS ticken die Uhren leider etwas anders. Dort wird das PHP mittels *FPM* dem sogenannten *FastCGI Process Manager* eingebunden. Das FPM stellt eine eigenen Dienst bereit, der die Anfragen annimmt, verarbeitet und das Ergebnis zurückliefert. Dieser kann über Unix-Sockets oder TCP angesprochen werden.

Nach der Installation genügt es den PHP-FPM-Dienst zu starten und zu aktivieren und den Apache einmal neustarten damit PHP-Dateien verarbeitet werden können (siehe Listing 8.36). Die Konfiguration wird praktischerweise direkt mitgeliefert.

```
[root@centos ~]# systemctl start php-fpm.service
[root@centos ~]# systemctl enable php-fpm.service
[root@centos ~]# systemctl restart httpd
```

Listing 8.36 CentOS: PHP-Dienst starten und aktivieren

Wie Sie überprüfen können, ob Ihre PHP-Installation ordnungsgemäß funktioniert, zeigen wir Ihnen in Abschnitt 8.3.3, »Funktionstest«.

nginx

Installieren Sie zunächst die Pakete `php` und `php-fpm` aus den Paketquellen. Kontrollieren Sie anschließend ob der Dienst läuft:

```
$ systemctl status php*-fpm.service
* php8.1-fpm.service - The PHP 8.1 FastCGI Process Manager
   Loaded: loaded (/lib/systemd/system/php8.1-fpm.service; enabled; vendor
          preset: enabled)
   Active: active (running) since Wed 2022-12-28 17:10:22 UTC; 3min 37s ago
     Docs: man:php-fpm8.1(8)
  Process: 8917 ExecStartPost=/usr/lib/php/php-fpm-socket-helper install
           /run/php/php-fpm.sock >
 Main PID: 8914 (php-fpm8.1)
   Status: "Processes active: 0, idle: 2, Requests: 0, slow: 0, Traffic: 0req/sec"
    Tasks: 3 (limit: 2238)
  Memory: 7.2M
     CPU: 97ms
  CGroup: /system.slice/php8.1-fpm.service
          |-8914 "php-fpm: master process (/etc/php/8.1/fpm/php-fpm.conf)" "" >
          |-8915 "php-fpm: pool www" "" "" "" "" "" "" "" "" "" "" "" "" "" "" >
          `--8916 "php-fpm: pool www" "" "" "" "" "" "" "" "" "" "" "" "" "" "" >

[...]
```

Listing 8.37 Überprüfung, ob der PHP-Dienst läuft

Unter openSUSE und CentOS wird der Dienst noch nicht laufen, da dort noch ein paar mehr Konfiguration notwendig sind, die wir uns nun genauer anschauen.



Im Übrigen wird das Asterisk im Kommando `systemctl status` über die Shell expandiert, so dass das Kommando auf allen PHP-Versionen funktioniert und sogar auf die generische Version – also zum Beispiel `php7-fpm.service`, `php8-fpm.service` oder `php-fpm.service`.

openSUSE

Nach der Installation läuft der Dienst nicht und kann auch nicht gestartet werden, da er zunächst konfiguriert werden muss. Dafür genügt es, die Standardkonfigurationen mit den zwei Befehlen aus Listing 8.38 zu aktivieren.

```
leap:~ # mv /etc/php8/fpm/php-fpm.conf.default /etc/php8/fpm/php-fpm.conf
leap:~ # mv /etc/php8/fpm/php-fpm.d/www.conf.default /etc/php8/fpm/php-fpm.d/www.conf
```

Listing 8.38 Anpassung in »`/etc/php-fpm.d/www.conf`«

Nun kann der Dienst gestartet und aktiviert werden:

```
leap:~ # systemctl start php-fpm.service
leap:~ # systemctl enable php-fpm.service
```

Listing 8.39 PHP starten und aktivieren unter openSUSE

CentOS

Nach der Installation müssen Sie nur noch den Benutzer und die Gruppe, unter denen der Dienst laufen soll, in der Konfigurationsdatei `/etc/php8/fpm/php-fpm.d/www.conf` korrigieren (siehe Listing 8.40).

```
[www]
...
; RPM: apache user chosen to provide access to the same directories as httpd
user = nginx
; RPM: Keep a group allowed to write in log dir.
group = nginx
```

Listing 8.40 Anpassung in »`/etc/php-fpm.d/www.conf`«

Die Werte stehen Standardmäßig auf `apache`, nach der Korrektur kann der Dienst gestartet und aktiviert werden:

```
[root@centos ~]# systemctl start php-fpm.service
[root@centos ~]# systemctl enable php-fpm.service
Created symlink /etc/systemd/system/multi-user.target.wants/php-fpm.service \
--> /usr/lib/systemd/system/php-fpm.service.
```

Listing 8.41 PHP starten und aktivieren unter CentOS

8.3.2 PHP in den Webseitenkonfigurationen aktivieren

Nun verfügt Ihr System zwar über PHP, allerdings müssen Sie diesen Umstand auch der Webseitenkonfiguration verraten. Darum kümmern wir uns in diesem Abschnitt.

Debian/Ubuntu

Bei Debian- und Ubuntu-Systemen wird bereits eine lauffähige Konfiguration mitgeliefert, so dass Sie Ihre Webseitenkonfiguration lediglich um den Inhalt auf Listing 8.42 erweitern müssen.

```
location ~ /\.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/var/run/php/php-fpm.sock;
}
```

Listing 8.42 Debian/Ubuntu: »/etc/nginx/sites-enabled/example.com«

Fügen Sie die Zeilen einfach unterhalb der DocumentRoot-Lokation ein. Darüber wird Ihrem nginx mitgeteilt, dass er alle Dateien mit der Suffix `php` an den Unix-Socket des PHP-FPM weitergeben soll – zusätzlich wird vorab die Standardparameter aus der Datei `/etc/nginx/snippets/fastcgi-php.conf` eingebunden. Nach einem Neuladen des Dienstes mittels `systemctl reload nginx` werden PHP-Dateien von Ihrem System interpretiert.

Centos

Bei CentOS-Systemem werden die Standardvariablen als Konfigurationsdatei mitgeliefert und eine Standardkonfiguration für PHP erstellt (siehe Listing 8.43).

```
# pass the PHP scripts to FastCGI server
#
# See conf.d/php-fpm.conf for socket configuration
#
index index.php index.html index.htm;

location ~ /\.(php|phar)(/.*)?$ {
    fastcgi_split_path_info ^(.+\.(?:php|phar))(/.*)$;

    fastcgi_intercept_errors on;
    fastcgi_index index.php;
    include fastcgi_params;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    fastcgi_param PATH_INFO $fastcgi_path_info;
    fastcgi_pass php-fpm;
}
```

Listing 8.43 Centos: Inhalt von »/etc/nginx/default.d/php.conf«

Darüber wird Ihrem nginx mitgeteilt, dass er alle Dateien mit der Suffix `php` oder `phar` an den Unix-Socket des PHP-FPM weitergeben soll – zusätzlich werden die Standardvariablen aus der Datei `/etc/nginx/fastcgi_params` eingebunden und weitere Standardkonfiguration gesetzt. Über den Befehl `include` können Sie diese Standardkonfiguration einfach in Ihre Webseitenkonfiguration aufnehmen, wie in Listing 8.44 dargestellt.

```
server {
    listen      80;
    server_name example.com www.example.com;
    root        /var/www/example.com;
    include     default.d/php.conf;

    location / {
        index index.html index.htm;
    }
}
```

Listing 8.44 Centos: »`/etc/nginx/sites-enabled/example.com`«

Auch wenn in der Standardkonfiguration der Parameter `index` enthalten ist, müssen Sie diesen in der Root-Lokation erneut setzen, wenn `index.php` direkt aufgerufen werden soll. Passen Sie die Zeile dafür so an »`index index.html index.html index.php`«.

openSUSE

Bei openSUSE-Systemen müssen Sie zwei Dinge beachten:

1. Zugriff via TCP und nicht via Socket
2. Apparmor

Entgegen der bisher vorgestellten Konfigurationen werden die Anfragen, die durch PHP verboten werden sollen, auf openSUSE-Systemen via TCP an den Port 9000 des `localhost` weitergeben und nicht via Unix-Sockets (siehe Listing 8.45). Daher ist die Konfiguration nahezu identisch z.B. zu der von CentOS – die Abweichung haben wir Ihnen in Fettschrift ausgewiesen:

```
server {
    listen      80;
    server_name example.com www.example.com;
    root        /var/www/example.com;

    location / {
        index index.html index.htm;
    }
}
```



```

location ~ \.(php|phar)(/.*)?$ {
    fastcgi_split_path_info ^(.+\.(?:php|phar))(/.*)$;

    fastcgi_intercept_errors on;
    fastcgi_index   index.php;
    include         fastcgi_params;
    fastcgi_param   SCRIPT_FILENAME    $document_root$fastcgi_script_name;
    fastcgi_param   PATH_INFO          $fastcgi_path_info;
    fastcgi_pass    127.0.0.1:9000;
}
}

```

Listing 8.45 openSUSE: »etc/nginx/sites-enabled/example.com«

Zusätzlich verhindert Apparmor die Ausführung von PHP-Dateien, da von nginx kein passender Regelsatz mitgeliefert wird. Um den Zugriff zu gewähren, können Sie entweder den gesamte PHP-Schutz mittels `aa-complain /usr/sbin/php-fpm` deaktivieren oder den Zugriffsschutz für `/srv/www/` als lokale Regel ergänzen – wie Sie dies umsetzen und was es sonst zu Apparmor zu sagen gibt, finden Sie in Kapitel 30.3, »Selbstabsicherung: AppArmor«.

8.3.3 Funktionstest

Um zu überprüfen, das PHP wirklich funktioniert, können Sie einfach in Ihrem Document-Root-Verzeichnis (im Beispiel bisher `/var/www/example.com`) die Datei `info.php` mit dem Inhalt aus Listing 8.46 anlegen.

```
<?PHP phpinfo(); ?>
```

Listing 8.46 Inhalt der Datei »info.php«

Wenn Sie diese Datei aufrufen und Ihre PHP-Installation einwandfrei funktioniert, dann werden Ihnen alle Informationen Ihrer PHP-Installation tabellarisch dargestellt. Anderenfalls erhalten Sie eine Fehlermeldung oder die PHP-Datei wird heruntergeladen, was stets ein Indiz dafür ist, das der PHP-Interpreter nicht gefunden und gestartet wurde.

8.3.4 Tipps und Tricks

Abschließend wollen wir Ihnen noch ein paar Tipps und Tricks mit an die Hand geben:

Commandline

PHP-Dateien müssen nicht ausschließlich im Browser ausgeführt werden. Sie können beliebige PHP-Dateien auch auf der Kommandozeile ausführen. Dafür müssen Sie lediglich das Paket `php-cli` installieren.

In nachstehenden Listing legen wir die Datei *time.php* an, in der lediglich die aktuelle Zeit als Unix-Timestamp ausgegeben wird, und führen diese auf der Kommandozeile aus:

```
root@ubuntu:~# echo '<?PHP echo time(); ?>' > time.php
root@ubuntu:~# php time.php
1672313437
```

Listing 8.47 PHP auf der Kommandozeile

Größenbeschränkung

Oftmals werden in PHP-Skripten größere Dateien oder Formulardaten verarbeitet. Standardmäßig lässt PHP aber nur Dateiupload mit einer Größe von 2 MB (Parameter `upload_max_filesize`) und Formulare mit insgesamt 8 MB (Parameter `post_max_size`) zu – das ist natürlich für viele Anwendungszwecke schlicht zu klein. Zusätzlich wird über den Parameter `memory_limit` die maximale Gesamtgröße des Hauptspeichers angegeben – Standardmäßig liegt es bei 128MB (siehe Listing 8.48). Wenn Sie größere Uploads oder Formulare zulassen wollen, müssen Sie unbedingt beachten, dass das `memory_limit` stets größer als die anderen Werte ist! Im Übrigen müssten Sie dieses auch erweitern, wenn Sie bei der Verarbeitung von großen Datenmengen, zum Beispiel aus einer Datenbank, den Fehler »PHP: Fatal Error: Allowed Memory Size of <WERT> Bytes Exhausted« erhalten.

```
; Maximum amount of memory a script may consume
; https://php.net/memory-limit
memory_limit = 128M
[...]

; Maximum size of POST data that PHP will accept.
; Its value may be 0 to disable the limit. It is ignored if POST data reading
; is disabled through enable_post_data_reading.
; https://php.net/post-max-size
post_max_size = 8M
[...]

; Maximum allowed size for uploaded files.
; https://php.net/upload-max-filesize
upload_max_filesize = 2M
```

Listing 8.48 PHP-Upload-Limits in »php.ini«



Größenangaben

Sie können die Werte im übrigen in den Einheiten K für kilo, M für mega und G für Giga angeben. Sie müssen also nicht 1024K für ein Megabyte angeben, dafür genügt die Angabe von 1M.

Diese Konfiguration wird in der Datei *php.ini* vorgenommen. Sie finden sie in der Regel unterhalb des Verzeichnisses */etc/php/* (openSUSE: */etc/php8/*). Dort wird pro Version ein eigenes Verzeichnis erstellt und darin wiederum werden INI-Dateien pro Instanz aufbewahrt. So finden Sie dort zum Beispiel eine für Apache2 (*/etc/php/8.1/apache2/php.ini*), eine weitere für die CLI (*/etc/php/8.1/cli/php.ini*) und sofern Sie es installiert haben auch eine für FPM. Achten Sie daher genau darauf die korrekte Datei anzupassen. CentOS verfolgt einen anderen Ansatz, dort gibt es nur eine INI-Datei und zwar */etc/php.ini*.

Info. Info? Info!

Das gebündelte Wissen zu PHP finden Sie auf der hervorragend gepflegten Webseite des Projekts unter *php.net*. Wenn Sie also Informationen zu einzelnen Funktionen, Konfigurationen oder Parameter suchen, dann lohnt sich ein Besuch dieser Webseite definitiv!

8.4 Fortgeschrittene TLS-Konfiguration und Sicherheitsfunktionen

Im Verlauf dieses Kapitels haben wir Ihnen bereits vorgestellt, wie Sie HTTPS in den Webservern aktivieren und konfigurieren. In diesem Abschnitt möchten wir Ihnen das Rüstzeug an die Hand geben, um Ihre Konfiguration sicher zu gestalten und dies auch zu überprüfen. Darüberhinaus stellen wir Ihnen ein paar weitere Sicherheitsfunktionen vor, mit denen Sie den Betrieb Ihrer Webanwendungen noch weiter absichern können.

8.4.1 SSL/TLS

Die *Transport Layer Security (TLS)*, ehemals *Secure Socket Layer (SSL)*, ermöglicht die sichere Übertragung von Informationen aus der Anwendungsschicht (z.B. HTTPS, FTPS oder IMAPS) über TCP/IP. Bei einem Verbindungsaufbau tauschen Server und Client Ihre Möglichkeiten aus und einigen sich auf die stärkste gemeinsame Alternative. In diesem Abschnitt zeigen wir Ihnen wie Sie dies für die Webserver Apache2 und nginx sicher einrichten, was Sie dabei beachten müssen und wie Sie Ihre Konfiguration überprüfen können.

Theorie

Zunächst ein wenig Theorie vorab. Generell wird TLS in Protokollen unterschieden, zum Beispiel:

- ▶ SSLv3
- ▶ TLS 1.0
- ▶ TLS 1.1
- ▶ TLS 1.2
- ▶ TLS 1.3

Die Versionen beschreiben jeweils die gültigen Verarbeitungsarten, die Methoden für den Auf- und Abbau einer Verbindung, die Schlüsselaustauschmethoden und weitere Rahmenbedingungen. Jedes Protokoll besteht im wesentlichen aus den folgenden Elementen (die Summe dieser Elemente wird im Übrigen als *Cipher-Suite* bezeichnet):

- ▶ Schlüsseleinigung/-austausch (z.B.: *RSA, DH (auch ADH, ECDH), PSK, SRP*)
- ▶ Verschlüsselung (z.B.: *RC4, DES, 3DES, AES*)
- ▶ Betriebsmodus (z.B.: *CBC, GCM*)
- ▶ Hashfunktion (z.B.: *MD5, SHA*)



Mehr dazu ...

Mehr zum Thema Kryptografie finden Sie in Kapitel 31, »Verschlüsselung und Zertifikate«.



Bitte beachten Sie, dass gerade die Header-Konfigurationen, die wir hier vorstellen, einige Webanwendungen kaputt machen können – falls nach der Aktivierung irgendetwas nicht so läuft wie erwartet, sollten Sie diese Konfiguration als erstes deaktivieren, um den Fehler einzugrenzen.

Neben den bereits gezeigten Möglichkeiten gibt es noch eine weitere Option, die Sie stets umsetzen sollten: das Abschalten der Server-Signatur. Über diese kann ein Angreifer wertvolle Informationen abgreifen, nämlich welche Software-Version Sie einsetzen. So kann leicht recherchiert werden, welche Angriffsvektoren existieren und welche Sicherheitslücken bereits bekannt sind. Für den Apache genügt es, wenn Sie die Parameter *ServerSignature Off* und *ServerTokens Prod* setzen. Bei Debian und Ubuntu finden Sie die Konfiguration in der Datei */etc/apache2/conf-enabled/security.conf*. Bei CentOS ist dies bereits vorgegeben (einkompliliert) und wird nicht zentral konfiguriert. Bei openSUSE ist es ebenfalls vorgegeben, allerdings in der Konfigurationsdatei */etc/apache2/global.conf*. Bei nginx müssen Sie den Parameter *server_tokens off* setzen. Bei Debian und Ubuntu ist die entsprechende Konfigurationszeile bereits auskommentiert in der *nginx.conf* vorhanden. Bei CentOS und openSUSE müssen Sie diese im http-Block ergänzen.

8.4.2 Konfiguration in Apache2

Neben der reinen TLS-Konfiguration können Sie noch viele weitere Parameter einsetzen, um den Betrieb weiter abzusichern. In Listing 8.49 sehen Sie dazu eine Beispielfonfiguration.

```
### SSL-Konfiguration
SSL Engine on
SSLCertificateFile /etc/apache2/ssl/certfile.crt
# SSLCertificateKeyFile /etc/apache2/ssl/keyfile.key
```

```

SSLOpenSSLConfCmd DHParameters /etc/apache2/ssl/dhparam.pem
SSLProtocol -all +TLSv1.3 +TLSv1.2
SSLOpenSSLConfCmd Curves X25519:secp521r1:secp384r1:prime256v1
SSLCompression off
SSLHonorCipherOrder on
SSLCipherSuite ECDH+AESGCM:EDH+AESGCM

### HSTS
Header always set Strict-Transport-Security "max-age=15768000; preload"

### X-Frame/CSS/XSS [OPTIONAL]
Header always set X-Frame-Options DENY
Header always set X-Content-Type-Options nosniff
Header always set X-XSS-Protection "1; mode=block"

```

Listing 8.49 Beispielkonfiguration »Apache2«

Bevor Sie die Konfiguration aus Listing 8.49 einspielen, müssen Sie zunächst die Datei *dhparam.pem* mit dem Befehl `openssl dhparam -out /etc/apache2/ssl/dhparam.pem 2048` erzeugen. Die Konfigurationen habe im Übrigen folgende Funktion:

- ▶ `SSLEngine on` = Aktiviert TLS/SSL
- ▶ `SSLCertificateFile <FILE>` = Zertifikat
- ▶ `SSLCertificateKeyFile <FILE>` = Wenn der Schlüssel separat gespeichert ist kann er der Speicherort so angegeben werden
- ▶ `SSLOpenSSLConfCmd DHParameters <FILE>` = Speicherort des Diffie-Hellmann Schlüssels für das *Perfect-Forward-Secrecy (PFS)* (sicherer Schlüsselaustausch)
- ▶ `SSLProtocol -all +TLSv1.3 +TLSv1.2` = Nur TLS-Protokolle 1.2 und 1.3 zulassen
- ▶ `SSLOpenSSLConfCmd Curves <DATA>` = Zugelassene Algorithmen für elliptische Kurven
- ▶ `SSLCompression off` = Abschaltend er SSL-Kompression (Verhinderung *CRIME*)
- ▶ `SSLHonorCipherOrder On` = Server- hat Vorrang zur Clientkonfiguration
- ▶ `SSLCipherSuite <CIPHER>` = Zugelassene Cipher-Suites
- ▶ `Header always set Strict-Transport-Security «VAL»` = HSTS aktivieren (einmal via HTTPS aufgerufene Seiten werden ausschließlich via HTTPS aufgerufen)
- ▶ `Header always set X-Frame-Options DENY` = Keine Einbindung von Inhalten via *IFrame*
- ▶ `Header always set X-Content-Type-Options nosniff` = Unterbindet das automatische Erkennen von Dateien durch den Browser
- ▶ `Header always set X-XSS-Protection "1; mode=block"` = Schutz vor *Cross-Site-Scripting-Attacks*

8.4.3 Konfiguration in nginx

Auch mit dem nginx können Sie neben der reinen TLS-Konfiguration viele weitere Parameter einsetzen, um den Betrieb weiter abzusichern. In Listing 8.50 geht es los.

```
### SSL-Konfiguration
ssl_certificate /etc/nginx/ssl/certfile.crt;
ssl_certificate_key /etc/nginx/ssl/keyfile.key;
ssl_dhparam /etc/nginx/ssl/dhparam.pem;

ssl_protocols TLSv1.2 TLSv1.3;
ssl_ecdh_curve brainpoolP512r1:brainpoolP384r1:secp384r1:brainpoolP256r1;
ssl_session_timeout 10m;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off;
ssl_prefer_server_ciphers on;
ssl_ciphers 'EECDH+AESGCM:EDH+AESGCM;';

### HSTS-Header
add_header Strict-Transport-Security "max-age=15768000; preload";

### X-Frame/CSS/XSS [OPTIONAL]
add_header X-Frame-Options DENY;
add_header X-Content-Type-Options nosniff;
add_header X-XSS-Protection "1; mode=block";
```

Listing 8.50 Beispielkonfiguration »nginx«

Auch beim nginx müssen Sie, bevor Sie die Konfiguration aus Listing 8.50 einspielen, zunächst die Datei *dhparam.pem* mit dem Befehl `openssl dhparam -out /etc/nginx/ssl/dhparam.pem 2048` erzeugen. Die Konfigurationen haben folgende Funktion:

- ▶ `ssl_certificate <FILE>` = Zertifikat
- ▶ `ssl_certificate_key <FILE>` = Schlüssel
- ▶ `ssl_dhparam <FILE>` = Speicherort des Diffie-Hellmann Schlüssels für das *Perfect-Forward-Secrecy (PFS)* (sicherer Schlüsselaustausch)
- ▶ `ssl_protocols TLSv1.2 TLSv1.3` = Nur TLS-Protokolle 1.2 und 1.3 zulassen
- ▶ `ssl_ecdh_curve <DATA>` = Zugelassene Algorithmen für elliptische Kurven
- ▶ `ssl_session_timeout 10m` = Ablaufzeit einer Session
- ▶ `ssl_session_tickets` = Deaktiviert die Wiederaufnahme einer Session
- ▶ `ssl_prefer_server_ciphers on` = Server- hat Vorrang zur Clientkonfiguration

- ▶ `ssl_ciphers <CIPHER>` = Zugelassene Cipher-Suites
- ▶ `add_header Strict-Transport-Security «VAL>` = HSTS aktivieren (einmal via HTTPS aufgerufene Seiten werden ausschließlich via HTTPS aufgerufen)
- ▶ `add_header X-Frame-Options DENY` = Keine Einbindung von Inhalten via *IFrame*
- ▶ `add_header X-Content-Type-Options nosniff` = Unterbindet das automatische Erkennen von Dateien durch den Browser
- ▶ `add_header X-XSS-Protection "1; mode=block"` = Schutz vor *Cross-Site-Scripting*-Attacken

8.4.4 Informationen und Anregungen

Informationen und Konfigurationsvorschläge schwirre Internet umher wie immer Sommer die Mücken am Wasser. Die einen sind veraltete, die anderen teilweise gefährlich und wiederum anderen top aktuell und umfassend erläutert.

Wir möchten Ihnen ein paar Links bereitstellen mit Quellen die wir blind weiterempfehlen können und die Sie bei der Anpassung und Kontrolle Ihrer TLS-Konfiguration durchaus konsultieren sollten – auch wenn Sie sie nur als Inspirationsquelle verwenden.

1. BSI

Das *Bundesamt für Sicherheit in der Informationstechnik (BSI)* hat sich in den vergangenen Jahren deutlich weiterentwickelt und verdient dem ihm anhaftenden Ruf der »Macher im Elfenbeinturm« nicht mehr. Die technische Richtlinie zur Kryptografie werden mindestens jährlich oder anlassbezogen aktualisiert und bieten eine guten Überblick. Folgende Dokumente empfehlen wir Ihnen als Pflichtlektüre:

- ▶ **TLS-Checkliste**
Eine kompakte Übersicht der wichtigsten Punkte: <https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TRO3116/TLS-Checkliste.pdf>
- ▶ **TR-02102-2: »Kryptographische Verfahren: Verwendung von Transport Layer Security (TLS)«**
Hintergrundinformationen und Erläuterungen: <https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TRO2102/BSI-TR-02102-2.html>
- ▶ **TR-03116-4: »Kryptographische Vorgaben für Projekte der Bundesregierung«**
Explizite Vorgaben für Projekte des Bundes – allerdings eine guter Überblick auch für die Absicherung anderer Kommunikation als HTTPS: <https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TRO3116/BSI-TR-03116-4.html>

2. »cipherlist.eu«

Das Projekt *cipherlist* wurde leider abgeschaltet, glücklicherweise haben sich ein paar fleißige Hände gefunden die die Seite unter anderem Namen fortführen. Wir nutzen gerne die Europäische-Variante von <https://cipherlist.eu/>. Dort finde Sie eine Vielzahl von TLS-Konfi-

gurationen für die unterschiedlichsten Dienste wie Web-, Mail-, FTP-Server und viele mehr. Neben der reinen TLS-Konfiguration finden Sie dort auch viele nützliche weitere Konfigurationsoptionen.

3. »ssllabs.com«

Um Ihre TLS-Konfiguration zu überprüfen, können Sie den Dienst von *ssllabs.com* verwenden. Mit dem von der Firma *Qualys Inc.* kostenfrei bereitgestellten Tool überprüfen Sie Ihre Webserver-Konfiguration auf Herz und Nieren. Sie müssen lediglich die zu prüfende Webseiten angeben, etwas Geduld mitbringen und erhalten anschließend eine Benotung nach dem amerikanischen System, bei dem ein *A+* das Pendant zur deutschen *1+* darstellt und entsprechend ein *F* die glatte *6* ist. Zusätzlich erhalten Sie eine ausführliche Zusammenfassung über die eingesetzten Zertifikaten (und deren Parameter), die TLS-Konfiguration nebst unterstützten Protokollen, angebotenen Cipher-Suites, eine Auflistung, welche Browser Ihre Webseite öffnen können und die Protokolldetails zu bekannten Attacke.

Kapitel 9

FTP-Server

Totgesagte leben länger – FTP wird allen Unkenrufen zum Trotz immer noch gerne und viel benutzt.

Eines der ältesten Protokolle für Dateiübertragungen hat auch in der heutigen Zeit noch seine Daseinsberechtigung. FTP-Server sind einfach installiert, das Protokoll hat einen geringen Overhead, und die Datenübertragung ist verglichen mit anderen Protokollen relativ schnell. Gerade im Vergleich mit Webservern (siehe Kapitel 8, »Webserver«) kann auch auf Maschinen mit geringer Hardwareausstattung oder auf virtuellen Servern mittels FTP eine große Anzahl an Kunden bedient werden.

9.1 Einstieg

Die konzeptionellen Schwächen von FTP lassen sich mittlerweile durch Erweiterungen des FTP-Protokolls ausräumen. Konfigurativen Herausforderungen an eine Firewall kann man durch Verwendung des passiven Modus begegnen. Sicherheitsbedenken – das Passwort wird im Klartext übertragen – werden durch *FTPS* (FTP über SSL, nicht zu verwechseln mit *SFTP*, was zur Secure Shell gehört) ausgeräumt.

9.1.1 Das File Transfer Protocol

Für FTP werden von der IANA¹ zwei Ports für die Kommunikation reserviert. Das sind Port 21 (TCP), der auch kurz der *FTP-Port* oder *Kommandokanal* genannt wird, und Port 20 (TCP), der als *FTP-Data* oder *Datenkanal* bekannt ist.

Aktives FTP

Beim aktiven FTP entscheidet der Client darüber, über welchen Port die Dateiübertragung laufen soll. Beim Verbindungsaufbau öffnet er einen zufälligen Port jenseits von 1023 und teilt diesen und die eigene IP-Adresse dem Server auf Port 21 mit. Das hört sich zunächst einmal widersinnig an, aber der Client könnte damit auch einen anderen FTP-Server angeben, an den die angeforderten Dateien gesendet werden sollen (FXP – »File Exchange Protocol«).

1 <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>

Bei Beginn der Übertragung öffnet der Server von seinem Port 20 eine Verbindung zu der vom Client angegebenen IP-Adresse und zu dem angegebenen Port. Da der komplette Dialog über zwei verschiedene Verbindungen läuft, können Client und Server während der Datenverbindung noch Kommandos austauschen.

Passives FTP

Im Falle einer passiven FTP-Übertragung sendet der Client das EPSV- oder PASV-Kommando, um dem Server mitzuteilen, dass ein passiver Transfer erfolgen soll. Jetzt öffnet der Server einen Port jenseits von 1023 und überträgt diesen mit der IP-Adresse an den Client. Passives FTP wurde nötig, weil viele Clients hinter Routing oder NAT »versteckt« sind und nur die IP-Adresse des Routers von außen erreicht werden konnte.

9.1.2 vsftpd

Wir haben uns für *vsftpd* (*very secure ftp daemon*) – siehe dazu <https://security.appspot.com/vsftpd.html> – entschieden, da er schnell und einfach zu konfigurieren ist. Laut Aussage des Entwicklerteams ist *vsftpd* sicherer als andere FTP-Server. Er wird mit dem Paketmanagement der Distribution installiert.

Die zentrale Konfigurationsdatei ist die */etc/vsftpd.conf* (CentOS: */etc/vsftpd/vsftpd.conf*), die in einer sehr gut dokumentierten Beispielfassung mit installiert wird. Obwohl die von uns betrachteten Distributionen *vsftpd* in der gleichen Version 3.0.5 – Debian bietet leider nur Version 3.0.3 – ausliefern, starten wir jedes Beispiel mit einer leeren Datei.

9.2 Download-Server

Ein Download-Server zeichnet sich dadurch aus, dass jeder User ohne Anmeldung von diesem Server Daten herunterladen kann. In diesem Zusammenhang wird auch häufig von *Anonymous FTP* gesprochen. An einem üblichen FTP-Server, der für anonymen Zugriff konfiguriert ist, kann man sich mit dem Usernamen »anonymous« oder »ftp« anmelden. Das Passwort kann bei der Anmeldung leer gelassen werden.

Zumeist ist es so, dass der Server nach der Anmeldung ein paar Informationen darüber gibt, was auf ihm zu finden ist. Standardmäßig zeigen FTP-Server Zeiten und Daten in *GMT* an, allerdings ist es üblich, dass man hier auf lokale Zeit umschaltet. All das leistet die folgende minimale Konfigurationsdatei:


```
listen=YES
listen_ipv6=NO
anonymous_enable=YES
dirmessage_enable=YES
```

```
ftpd_banner=Willkommen auf meinem ftp-Server.
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
```

Listing 9.1 Konfigurationsdatei für den Download-Server

Option	Bedeutung
listen=YES	Der Server läuft als eigenständiges Programm und nicht in Verbindung mit (x)inetd.
listen_ipv6=NO	Der Server horcht auf Anfragen via IPv6. Bitte beachten Sie auch die nachfolgende Warnung.
anonymous_enable=YES	Anonyme Benutzung ist erlaubt.
dirmessage_enable=YES	Beim Wechsel in Verzeichnisse werden <i>.message</i> -Dateien angezeigt, die den Inhalt beschreiben.
ftpd_banner=Text	Der Text erscheint direkt nach dem Verbindungsaufbau und vor dem Login-Dialog.
use_localtime=YES	Das Datum der Dateien wird in lokaler Zeit und nicht in GMT ausgegeben.
xferlog_enable=YES	Datentransfers werden protokolliert.
connect_from_port_20=YES	Datentransfers werden von Port 20 des Servers initiiert, so wie es der Standard empfiehlt.

Tabelle 9.1 Optionen für »Anonymous FTP«

Anders als in Version 2 des vsftpd dürfen die Konfigurationsoptionen `listen` und `listen_ipv6` nicht gleichzeitig gesetzt sein. 

Die Daten finden sich im Verzeichnis `/srv/ftp` – bei CentOS ist es das Verzeichnis `/var/ftp`. In der Datei *.message* können Sie je Verzeichnis eine Nachricht hinterlegen, die immer dann angezeigt wird, wenn der User in dieses Verzeichnis wechselt.

Ein Beispiel für diese Datei finden Sie in Listing 9.2:

Im Verzeichnis `Incoming` finden Sie die unsortierten Eingänge.

Listing 9.2 Eine *.message*-Musterdatei

Ein Neustart mittels `systemctl restart vsftpd` ist nicht nötig. Eine Beispielanmeldung könnte folgendermaßen aussehen:

```
user@client:~# ftp server
Connected to server.
220 Willkommen auf meinem ftp-Server.
Name (server:user): anonymous
331 Please specify the password.
Password:
230-Im Verzeichnis Incoming finden Sie die unsortierten Eingänge.
230-
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> quit
221 Goodbye.
```

Listing 9.3 Beispielanmeldung auf unserem FTP-Server



Wir empfehlen es Ihnen ausdrücklich, die Option `anon_upload_enable` für Server, die im Internet stehen, nicht zu setzen. Die rechtlichen Konsequenzen, die durch Missbrauch entstehen können, sind nicht absehbar.

9.3 Zugriff von Usern auf ihre Homeverzeichnisse

Ein weiterer Anwendungszweck für FTP-Server ist es, Benutzern die Möglichkeit zu geben, Dateien in ihr Homeverzeichnis hochzuladen. Um dieses Ziel zu erreichen, müssen wir die Konfigurationsdatei um wenige Zeilen erweitern. Einige Erläuterungen zu Listing 9.4 finden Sie in Tabelle 9.2.

```
[...]
listen=YES
anonymous_enable=YES
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
# Lokale Nutzer ab hier
local_enable=YES
write_enable=YES
local_umask=022
chroot_local_user=YES
pam_service_name=vsftpd
local_root=/home
```

Listing 9.4 Konfigurationsdatei für den Zugriff von lokalen Benutzern auf ihre Homeverzeichnisse

Option	Bedeutung
local_enable=YES	Der Zugriff von lokal definierten Usern wird erlaubt.
write_enable=YES	Diese User haben Schreibzugriff.
local_umask=022	Die <i>umask</i> für neue Dateien ist 022 (sonst wäre sie 077).
chroot_local_user=YES	Lokal definierte User dürfen via FTP ausschließlich auf ihr Homeverzeichnis zugreifen. Mehr Informationen zu <i>chroot</i> finden Sie in Abschnitt 30.2, » <i>chroot</i> «.
pam_service_name=vsftpd	Hier geben Sie an, mit welchem PAM-Service die Authentifizierung erfolgen soll.
local_root=/home	Das <i>root</i> -Verzeichnis darf nicht für den User schreibbar sein. In diesem Fall darf der FTP-User auch auf die Daten anderer User zugreifen, wenn die Rechte es erlauben.

Tabelle 9.2 Optionen für den Zugriff auf die Homeverzeichnisse

9.4 FTP über SSL (FTPS)

Eine der größten Schwachstellen von FTP ist, dass Username und Passwort im Klartext übertragen werden. Diesen Umstand räumt eine Erweiterung des Protokolls auf *FTPS* – FTP über SSL – aus. Hiermit werden sowohl der Benutzername und das Passwort wie auch der eigentliche Datentransfer verschlüsselt. Die nun folgenden Befehle erklären im Schnelldurchgang den Weg zu einem eigenen Zertifikat. Ausführliche Informationen zu Verschlüsselung und Zertifikaten finden Sie in Kapitel 31, »Verschlüsselung und Zertifikate«.

Der Befehl aus Listing 9.5 erstellt das Zertifikat und den Schlüssel in der Datei *vsftpd.pem*. Diese Datei kopieren Sie bitte nach */etc/ssl/private/vsftpd.pem*; im Fall von CentOS müssen Sie das Verzeichnis erst erstellen.

```
/usr/bin/openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout vsftpd.pem \
-out vsftpd.pem
```

Listing 9.5 Zertifikat erzeugen

Wenn Sie einen »offiziellen« FTP-Server betreiben, ist es in jedem Fall sinnvoll, ein korrektes Zertifikat zu verwenden, das von einer bekannten Zertifizierungsstelle unterschrieben wurde.

Die eigentliche Konfigurationsdatei, um verschlüsselte Verbindungen zu ermöglichen, besteht nur aus wenigen zusätzlichen Befehlen:



```
listen=YES
anonymous_enable=YES
dirmessage_enable=YES
ftpd_banner=Willkommen auf meinem FTP-Server.
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
# Lokale Nutzer ab hier
local_enable=YES
write_enable=YES
local_umask=022
chroot_local_user=YES
pam_service_name=vsftpd
local_root=/home

# Verschlüsselung ab hier
ssl_enable=YES
allow_anon_ssl=YES
force_anon_data_ssl=YES
force_anon_logins_ssl=YES
force_local_data_ssl=YES
force_local_logins_ssl=YES
rsa_cert_file=/etc/ssl/private/vsftpd.pem
```

Listing 9.6 Konfigurationsdatei, um verschlüsselte Verbindungen zu ermöglichen

Option	Bedeutung
ssl_enable=YES	Basisoption, um überhaupt eine Verschlüsselung durchführen zu können
allow_anon_ssl=YES	Verschlüsselung des anonymen Zugangs
force_anon_data_ssl=YES	Verschlüsselung des Datentransfers
force_anon_logins_ssl=YES	Verschlüsselung des Logins
force_local_data_ssl=YES	Datenverschlüsselung für lokal angelegte Benutzer
force_local_logins_ssl=YES	Verschlüsselung des Anmeldevorgangs
rsa_cert_file=/etc/ssl/private/vsftpd.pem	Speicherplatz des verwendeten Serverzertifikats

Tabelle 9.3 Optionen für »Anonymous FTP« bei verschlüsselten Verbindungen

Das Beispiel aus Listing 9.6 (siehe auch die Erläuterungen in Tabelle 9.3) verschlüsselt alle Verbindungen, auch die, die von einem anonymen User gestartet werden.

Über den Sinn oder Unsinn lässt sich streiten. Der anonyme Zugriff kann von jedem verwendet werden, dazu sind weder ein spezieller User noch ein Passwort nötig. Allerdings kann man von außen nicht feststellen, welche Datei übertragen wurde, wenn die Verbindung verschlüsselt war.

Verschlüsselung benötigt Rechenzeit. Daher sind gerade bei den anonymen Benutzern der Sicherheitsaspekt, dass man nicht herausbekommen kann, welche Datei übertragen wurde, und der Ressourcenhunger gegeneinander abzuwägen.

9.5 Anbindung an LDAP

Die Anbindung an LDAP ist einfach. Bitte installieren Sie für openSUSE den LDAP-Client via YaST, bei CentOS installieren Sie *openldap-clients* und bei Debian und Ubuntu die Pakete *libnss-ldap* und *libpam-ldap*.

Wenn Sie nach der Installation des Clients in der Datei */etc/nsswitch.conf* die Option *ldap* zur *passwd*- und *group*-Zeile – wie im folgenden Auszug – hinzufügen und den *Name Service Cache Daemon* mittels `systemctl restart nscd` neu starten, benutzt *vsftpd* auch LDAP zur Authentifizierung.

```
passwd:      compat ldap
group:      compat ldap
```

Listing 9.7 Änderung an der Datei */etc/nsswitch.conf* zur Aktivierung von LDAP

Weitergehende Informationen zur Konfiguration von LDAP finden Sie in Abschnitt 17.4.6, »Konfiguration des LDAP-Clients«.

Kapitel 10

Mailserver

Dieses Kapitel zeigt Ihnen den sauberen Aufbau eines Mailrelays mit Postfix nebst grundlegendem Spam- und Virenschutz und erklärt, wie Sie Ihr Mailrelay vor einem Groupwareserver wie Exchange oder einem IMAP-Server wie Dovecot betreiben sollten. Außerdem zeigen wir Ihnen, wie Sie mit wenigen Schritten Dovecot als flexiblen und mächtigen IMAP-Server in Betrieb nehmen können.

E-Mail ist einer der ältesten Dienste im Internet überhaupt – und wird gleichzeitig von vielen Administratoren in seiner Komplexität unterschätzt. Mailserver sind mächtig und komplex, schließlich agieren sie nicht nur als Server, die auf eine konkrete Anfrage eines Clients reagieren, sondern arbeiten zugleich auch als autonom entscheidende Clients, die aufgrund eigener Logik andere Server kontaktieren und ihnen Mails zustellen. Es gibt extrem viele sinnvolle, vor allem aber auch sehr viele nicht sinnvolle Einstellungsmöglichkeiten.

Eigentlich sind Mailserver einfach. Zumindest, wenn man sich an die Regeln hält. Viele Administratoren arbeiten hier zu viel mit »Trial and Error«, glauben, es wäre egal, auf welchem Weg ein Ziel genau erreicht wird, und meinen, sie könnten an Schrauben so lange drehen, bis es dann irgendwie trickreich funktioniert. Doch genau hier werden die Komplexität der Zusammenhänge, die Auswirkungen vieler Kleinigkeiten unterschätzt. Konfigurieren Sie Mailserver immer recht penibel »im Sinne des Erfinders«, klar und stringent, und Sie werden feststellen, dass Mailserver auf einmal einfach, problemlos und pflegeleicht sein können.

10.1 Postfix

Postfix wurde 1997 von IBM zunächst unter den Namen *VMailer* und *Secure Mailer* entwickelt, um eine schnelle, sichere und bequem zu administrierende Alternative zum verbreiteten *Sendmail* zu bieten. Postfix-Autor Wietse Zwart hatte sich bereits viele Lorbeeren als Autor der TCP-Wrapper-Bibliothek (*/etc/host.allow* und */etc/host.deny*) und als Autor des Netzwerkscanners *Saint/Satan* erworben.

Als Spezialist für sichere Programmierung nutzte er die Gelegenheit und gab Postfix von vornherein ein sicheres Design und klare Entwicklungsprinzipien mit auf den Weg. Durch die Aufteilung in verschiedene, möglichst kleine, überschaubare Module konnte eine klare Aufgaben- und Rechtentrennung durchgesetzt werden, um Fehler und Angriffsflächen zu

minimieren. Wann immer es irgendwie geht, wird auf den Einsatz von root-Rechten verzichtet. Und das hat Erfolg: Im Gegensatz zu seinen Mitbewerbern ist für Postfix bis heute kein einziger root-Exploit bekannt geworden.¹

Postfix wird bis heute sehr aktiv weiterentwickelt und bietet auch bei neuen Entwicklungen wie DANE stets als eines der ersten Programme eine funktionierende Implementierung in mustergültiger Qualität.

10.1.1 Installation der Postfix-Pakete

Für das folgende Kapitel sind Sie nicht auf die allerneuesten Postfix-Versionen angewiesen, sondern sollten ganz normal bei der Version bleiben, die Ihre Linux-Distribution mitbringt. Zu beachten ist lediglich, dass manche Distributionen Postfix sehr stark in Unterpakete aufgeteilt haben. So ist oft die Unterstützung für LDAP, MySQL, PostgreSQL, aber auch PCRE-Pattern in eigene Unterpakete aufgeteilt. Schauen Sie also zunächst mittels

- ▶ `apt-cache search` (Debian/Ubuntu),
- ▶ `dnf search` (CentOS) oder
- ▶ `zypper search` (openSUSE)

nach, welche Pakete bei Ihnen vorhanden sind. Für dieses Kapitel reicht stets das Grundpaket, doch wenn Sie später beispielsweise auf eigene Faust mit einer Datenbankanbindung weitermachen, sollten Sie im Hinterkopf behalten, dass Sie gegebenenfalls weitere Unterpakete installieren müssen.

10.1.2 Grundlegende Konfiguration

Ein frisch installierter Postfix-Server ohne besondere Konfigurationsanpassungen ist an sich bereits ein lauffähiger, funktionierender Mailserver. Mails aus dem Internet nimmt er an, wenn sie an seinen Hostnamen adressiert sind und er die Empfänger in `/etc/passwd` vorfindet. Gleichzeitig akzeptiert er Mails seiner lokalen Nutzer und von Hosts in seinem eigenen Subnetz und sendet sie auch an externe Ziele ab, die er über DNS-MX-Records ermitteln kann. Die Konfiguration von Postfix spielt sich in `/etc/postfix` ab, namentlich in den beiden Dateien `main.cf` und `master.cf`. Dazu gesellen sich je nach Konfiguration zahlreiche weitere Hilfsdateien mit Domainlisten, Weiterleitungen oder White- und Blacklists, auf die durch entsprechende Parameter aus der `main.cf` verwiesen wird.

Bevor Sie mit der Konfiguration ans Eingemachte gehen, müssen Sie in Ihrer `main.cf` einige grundlegende Einstellungen prüfen und gegebenenfalls anpassen:

¹ Einzige Ausnahme: Es gab einen root-Exploit in der OpenSSL-Bibliothek, der alle Programme betraf, die gegen OpenSSL verlinkt waren, also auch Postfix.

- ▶ `inet_interfaces = all`

Diese Einstellung legt fest, auf welchen IP-Adressen Ihr Postfix »lauschen« soll. Per Default ist hier bei vielen Distributionen das Keyword `localhost` eingetragen, damit Desktop-Systeme oder normale Server nicht unbeabsichtigt mit einem offenen Port 25 im Internet erreichbar sind. Theoretisch könnten Sie hier (durch Kommata getrennt) alle IP-Adressen Ihres Systems auflisten – besser ist es jedoch in der Praxis, hier durch das Keyword `all` Postfix zunächst auf allen IP-Adressen (`*:25`) lauschen zu lassen und etwaige Einschränkungen auf bestimmte IP-Adressen erst später in der *master.cf* vorzunehmen.

```
# Desktops und Server sollten nicht aus dem Netz erreichbar sein:
inet_interfaces = localhost
```

```
# Produktivbetrieb mit offizieller IP-Adresse und localhost:
inet_interfaces = 80.70.60.50, localhost
```

```
# Idealerweise jedoch auf allen Interfaces Ports öffnen:
inet_interfaces = all
```

- ▶ `myhostname = mail.example.com`

Dies ist der Hostname des Postfix-Systems, mit dem es sich im SMTP-Protokoll zum Dienst meldet – und mit dem sich Postfix beim Versand auch anderen Mailservern gegenüber vorstellt. Der korrekt eingestellte Hostname ist für einen Mailserver extrem wichtig. Fehlkonfigurationen an dieser Stelle sind die häufigste und zugleich überflüssigste Fehlerquelle. Achten Sie stets darauf, dass `myhostname` und der Reverse Lookup Ihres Mailervers exakt übereinstimmen. Da Spam versendende Botnetze aus verschiedenen Gründen bei der Angabe ihres Hostnamens häufig lügen, prüfen Anti-Spam-Systeme die Übereinstimmung von Hostnamen und Reverse Lookup sehr penibel. Schlampfen Sie nun bei der sauberen Definition von Hostname und Reverse Lookup, werden Ihre E-Mails von anderen Servern unnötig häufig als Spam herausgefiltert. Postfix kennt auch den Parameter `mydomain`, doch sollten Sie diesen nicht ausdrücklich in der *main.cf* definieren. Postfix ermittelt `mydomain` automatisch, indem es den Hostanteil von `myhostname` weglässt.

- ▶ `mydestination = $myhostname, localhost.$mydomain, localhost`

In diese Zeile schreiben Sie alle Domainnamen, für die Ihr Postfix-Server lokal zuständig ist und deren Empfänger er als normale Systemnutzer in */etc/passwd* vorfindet. Man sagt dazu, dass Postfix die *final destination* für diese Domains ist. Am besten ist es, wenn Sie hier die Default-Konfiguration von Postfix beibehalten, nach der Postfix Mails ausschließlich für seine eigenen Hostnamen annimmt. Die Annahme von Mails an zusätzliche Domains klären wir später. Die Variablen `$myhostname` und `$mydomain` sollten dabei so in `mydestination` enthalten sein – Postfix ersetzt sie dann selbstständig. So ist sichergestellt, dass Postfix konsistent konfiguriert ist und sich `myhostname`, `mydomain` und `mydestination` nicht widersprechen.



Viele Anleitungen sind im Grundsatz falsch

Achtung: Sehr viele Anleitungen erklären zu `mydestination`, man solle hier auch alle Domains eintragen, die an nachgeordnete Systeme weitergeleitet werden, also für die Postfix nur ein Relay ist. Das ist jedoch falsch und führt zu zahlreichen Folgeproblemen, denn Postfix ist ja für diese Domains weder *final destination* noch findet er die zugehörigen Empfänger als lokale Shell-Nutzer vor.

Stattdessen gehören zu relayende Domains in den Parameter `relay_domains`, wie wir gleich ausführlicher beleuchten werden.

- `mynetworks = 127.0.0.0/8, [::1], 192.168.1.0/24`

Administratoren von Mailservern müssen immer aufpassen, dass ihre Server kein *Open Relay* sind, dass sie also nicht E-Mails von beliebigen externen IP-Adressen an ebenfalls externe Ziele annehmen. Ansonsten ist es stets nur eine Frage der Zeit, bis Spammer ein Open Relay nutzen, um Millionen von Spam-Mails auf diesem Weg bequem an die User zu versenden. Üblicherweise will man nur seinen eigenen IP-Adressen und Subnetzen einen Mailversand »nach draußen« erlauben.

Allerdings: Sie müssen den Parameter `mynetworks` in den meisten Fällen gar nicht ausdrücklich in die `main.cf` aufnehmen. Denn: Fehlt `mynetworks` in der `main.cf`, so bestimmt Postfix den Wert für `mynetworks` automatisch und setzt alle IP-Adressen des Hosts.²

Je nach Version und Paketierung kennt Postfix derzeit gut und gerne 800 Parameter, weil so gut wie alles bei Postfix über einen Parameter einstellbar ist. Alle diese Werte haben einen sinnvollen Default-Wert, sodass längst nicht alle in der `main.cf` stehen.

Dort finden Sie normalerweise nur diejenigen Parameter, die Sie abweichend vom Default-Wert tatsächlich anpassen wollen. In der Praxis werden das selbst in komplexeren Setups nur rund 20 bis 30 Optionen sein.

Über das Tool `postconf` können Sie jederzeit abfragen, welchen Wert ein Parameter aus Sicht von Postfix hat:

```
mail:~ # postconf myhostname
mail.example.com
mail:~ # postconf mynetworks
mynetworks = 127.0.0.0/8 192.168.200.0/24 10.0.40.6/32
```

Listing 10.1 »postconf« zeigt den Inhalt der Postfix-Parameter.

² Dies geschieht über den Parameter `mynetworks_style=host`, der bestimmt, wie `mynetworks` automatisch ermittelt werden kann. In älteren Postfix-Versionen war standardmäßig früher `mynetworks_style=subnet` aktiv, sodass alle IP-Subnetze des Postfix-Servers freigeschaltet wurden.



systemd versus syslog

In Sachen Logging verlassen sich viele Distributionen mittlerweile ganz auf *systemd* und dessen eigene Log-Funktionalitäten. Mit dessen Befehl `journalctl` können Sie gezielt alle Mailserver-Meldungen von Postfix und den später in diesem Kapitel vorgestellten Dovecot und Rspamd abfragen lassen:

```
journalctl -u postfix-.service -u dovecot -u rspamd --since "2021-01-10" \
  --until "2021-01-11 03:00"
```

Listing 10.2 Logdaten mit »journalctl« abfragen

Wenn Sie weiterhin auf den guten alten (*r*)*syslog* setzen wollen, finden Sie Ihr Logfile in `/var/log/mail` (openSUSE) bzw. `/var/log/mail.log` (Debian/Ubuntu). Bleiben diese Dateien leer, so prüfen Sie, ob (*r*)*syslog* bei Ihnen überhaupt installiert wurde, und holen Sie diese Installation kurzerhand nach (siehe Kapitel 12, »Syslog«).

10.1.3 Postfix als Relay vor Exchange, Dovecot oder anderen Backends

Damit Postfix auch als Relay vor anderen Servern E-Mails annehmen kann, müssen Sie noch die Parameter `relay_domains` und `transport_maps` setzen. Der erste sorgt dafür, dass Postfix Mails an die gewünschten Domains annimmt, der zweite sorgt dafür, dass Postfix diese Mails auch wieder an das gewünschte Ziel weiterleiten kann. Die Angabe von `transport_maps` ist hier ausnahmsweise notwendig, da Postfix sich in diesen Fällen nicht mehr auf die MX-Records im DNS verlassen kann, denn diese zeigen ja auf ihn selbst als das für das Internet zuständige Mailrelay.

Üblicherweise versenden Mailserver E-Mails über das SMTP-Protokoll auf Port 25. Auf der »letzten Meile« zwischen Mailrelay und Exchange- oder IMAP-Server kommt jedoch besser der kleine Bruder *Local Mail Transfer Protocol* (LMTP) auf Port 24 zum Einsatz. LMTP unterscheidet sich nur in kleinen Details von SMTP, es ist, wenn man so will, eine Art SMTP-Dialekt.

Mailserver können beim Einsatz von LMTP bei Mails an mehrere Empfänger gleichzeitig einfacher feststellen, ob die Mail in den verschiedenen Postfächern gespeichert werden konnte oder ob Quotas, Blacklisting oder andere Fehler dies verhindert haben. Legen Sie darum eine zweisepaltige Datei `/etc/postfix/relay_domains` an, in der Sie sowohl die Domains als auch das jeweilige Routing-Ziel auf Basis des LMTP- oder SMTP-Protokolls eintragen:

```
# Diese Domain wird an einen lokalen Dovecot/Cyrus-Server weitergegeben
example.com    lmtp:[127.0.0.1]
```

```
# Diese Domain wird an einen entfernt laufenden Exchange-Server geroutet
example.net    lmtp:[exchange.example.net]
```

```
# Diese Sub-Domain leiten wir per SMTP an den Mailinglistenserver weiter  
listen.example.org      smtp:[listenserver.example.org]
```

Listing 10.3 Die Syntax der kombinierten Relay-Domains- und Routing-Tabelle

Setzen Sie Hostnamen dabei stets in eckige Klammern, wenn Sie exakt diesen Mailserver, also die IP-Adresse hinter diesem Servernamen, ansprechen wollen. Lassen Sie die eckigen Klammern weg, würde Postfix zuerst prüfen, ob für diesen Server ausdrücklich MX-Records im DNS hinterlegt sind, und die Mails dann dorthin zustellen. Nur hilfsweise würde er sonst auf die A-Records dieser Hostnamen zurückgreifen.

Nun können Sie aus der *main.cf* mit `relay_domains` auf diese Tabelle verweisen und gleichzeitig Postfix über `transport_maps` anweisen, für Routing-Einträge neben der normalen `/etc/postfix/transport` auch alle Tabellen von `relay_domains` zu verwenden:

```
relay_domains = hash:/etc/postfix/relay_domains  
transport_maps = hash:/etc/postfix/transport, $relay_domains
```

Listing 10.4 Relay-Domains in Postfix einrichten

So haben Sie alle Routing-Fragen Ihrer zu relayenden Domains mit nur einem einzigen Eintrag in `/etc/postfix/relay_domains` geklärt. Die Datei `/etc/postfix/transport` benötigen Sie für Ihre eigenen Domains nicht – aber vielleicht möchten Sie hier in Ausnahmefällen einmal Routing-Einträge für Domains anderer Provider eintragen (deren E-Mails Sie deswegen ja nicht gleich als Relay aus dem Internet annehmen wollen).

Einige Distributionen haben `/etc/postfix/transport` noch nicht einmal als leere Datei vorbereitet. Wenn Sie über `transport_maps` auf eine nicht existente Datei verweisen, muss Postfix seine Arbeit einstellen. Legen Sie im Zweifelsfall `/etc/postfix/transport` als leere Datei an, und übersetzen Sie sie mit dem `postmap`-Kommando in eine Berkeley-Datenbank:

```
mail:~ # touch /etc/postfix/transport  
mail:~ # ls -la /etc/postfix/transport*  
-rw-r--r-- 1 root root    0 Mar 26 09:51 /etc/postfix/transport  
mail:~ # postmap /etc/postfix/transport  
mail:~ # ls -la /etc/postfix/transport*  
-rw-r--r-- 1 root root    0 Mar 26 09:51 /etc/postfix/transport  
-rw-r--r-- 1 root root 12288 Mar 26 09:52 /etc/postfix/transport.db
```

Listing 10.5 Relay-Domains in Postfix einrichten**Hash- und Btree-Tabellen erfordern ein »postmap«-Kommando**

Sobald Postfix in seiner Konfiguration angewiesen wird, über `hash:` oder `btree:` auf seine Tabellen zuzugreifen, müssen Sie nach jeder Änderung in der Datei das Kommando `postmap`

<filename> ausführen, also beispielsweise `postmap relay_domains`, `postmap transport` oder `postmap access_recipient`. Das `postmap`-Kommando wandelt die Dateien in eine kleine Berkeley-Datenbank mit der Endung `.db` um – und nur diese Datei wird von Postfix letzten Endes ausgewertet, auch wenn die `.db`-Endung nicht in der Konfiguration genannt ist.

10.1.4 Die Postfix-Restrictions: Der Schlüssel zu Postfix

Zu einem richtigen Mailserver gehört natürlich weit mehr als nur der Umstand, dass er Mails an bestimmte Domains von überall her annimmt bzw. Mails der eigenen IP-Bereiche an den Rest der Welt versenden kann. Spam-Schutzmaßnahmen, Empfängerprüfungen oder White-/Blacklists gehören zur elementaren Grundausstattung eines Mailservers und zur täglichen Arbeit eines Postmasters.

Ob eine Mail angenommen wird oder nicht, entscheidet sich dabei ganz wesentlich schon, bevor der eigentliche E-Mail-Inhalt übertragen wird. Bereits aus den Daten des SMTP-Protokolls lässt sich in vielen Fällen erkennen, dass eine Mail unerwünscht ist, die absendende IP-Adresse als Spam-Versender aufgefallen ist oder dass die Verbindung aus anderen Gründen als verdächtig gelten muss. All das entscheidet sich bei Postfix in den sogenannten *Restrictions*. Passend zu den fünf Stufen einer SMTP-Übertragung gibt es fünf Restrictions, die zu verschiedenen Zeitpunkten durchlaufen werden:

- ▶ die `smtpd_client_restrictions` nach dem Connect
- ▶ die `smtpd_helo_restrictions` nach dem HELO-Kommando
- ▶ die `smtpd_sender_restrictions` nach dem MAIL FROM:-Kommando
- ▶ die `smtpd_relay_restrictions` nach jedem RCPT TO:-Kommando
- ▶ die `smtpd_recipient_restrictions`, ebenfalls nach jedem RCPT TO:-Kommando

Viele Anleitungen erklären, dass an verschiedenen Stellen in den Restrictions nun umfangreiche Konfigurationsarbeiten notwendig seien: Sender-Blacklistings (`smtpd_sender_restrictions`), RBL-Abfragen (Real-time Blacklists)(`smtpd_client_restrictions`) und vieles andere mehr. Doch dieser Aufwand und diese Komplexität sind nicht nur überflüssig, sie verhindern in den meisten Fällen sogar eine gute Konfiguration. Denn ob eine Mail wirklich angenommen werden soll oder nicht, kann in manchen Fällen erst nach Bekanntgabe des Empfängers, also nach dem RCPT TO:-Kommando entschieden werden. So soll ein gesperrter Absender beispielsweise noch beim Helpdesk oder Postmaster um Rat fragen können. Insofern müssen Sie wissen, wer der Empfänger ist, bevor Sie entscheiden, ob Sie den Absender ablehnen wollen. Hinzu kommt, dass Postfix vorhandenes Wissen sowieso nicht vergisst. Auch in den `smtpd_recipient_restrictions` können noch Absender-Sperrlisten oder RBL-Abfragen über die Client-IP-Adresse ausgeführt werden. Auch aus dieser Sicht gibt es keinen Grund, »zu früh« zu prüfen.

Aufgrund eines falschen Grundverständnisses und auch halbgar zusammengestellter Postfix-HowTos hatten sich in der Praxis viele gefährliche Konfigurationsfehler in die Restrictions eingeschlichen. Mit der Postfix-Version 2.10 hat Postfix-Autor Wietse Zwartema darum vor die `smtpd_recipient_restrictions` noch die `smtpd_relay_restrictions` gestellt. Beide werden direkt nacheinander durchgeprüft. Insofern sind die `smtpd_relay_restrictions` eigentlich überflüssig, helfen Einsteigern jedoch, den Überblick zu behalten und die Konfiguration für ausgehende bzw. für eingehende Mails besser auseinanderhalten zu können. Eine Hilfe, die man nutzen kann, aber nicht nutzen muss. Wir empfehlen: Lassen Sie `smtpd_relay_restrictions` zu Ihrem eigenen Schutz auf dem Standardwert stehen.

Ansonsten ziehen wir es vor, alle gewünschten Prüfungen übersichtlich und logisch konsequent an einer einzigen Stelle zusammenzufassen: in den `smtpd_recipient_restrictions`. Hier kommt es entscheidend auf die richtige Reihenfolge an, damit sich keine Prüfungen widersprechen oder in der falschen Reihenfolge ausgeführt werden. Denn die Prüfungen werden der Reihe nach abgearbeitet: Sobald eine Prüfung ein eindeutiges positives Ergebnis (PERMIT – Mail wird angenommen) oder negatives Ergebnis (REJECT – Mail wird abgelehnt) liefert, wird die Entscheidung ausgeführt, und auf weitere Prüfungen in dieser Restriction kommt es dann nicht mehr an (*First match wins*). Aus unserer langjährigen Praxis empfehlen wir folgenden Aufbau:

```
# Musterlösung smtpd_recipient_restrictions
# Heinlein Support GmbH, http://www.heinlein-support.de

smtpd_recipient_restrictions =
# White-/Blacklistings für Empfänger, Sender, HELO und Client-IPs
    check_recipient_access hash:/etc/postfix/access_recipient
    check_sender_access hash:/etc/postfix/access_sender,
    check_helo_access hash:/etc/postfix/access_helo,
    check_client_access cidr:/etc/postfix/access_client,
# Keine Mails mit nicht existenten Absendern/Empfängern annehmen!
    reject_non_fqdn_sender,
    reject_non_fqdn_recipient,
    reject_unknown_sender_domain,
    reject_unknown_recipient_domain,
# Mails unserer Nutzer/Server erlauben!
    permit_sasl_authenticated,
    permit_mynetworks,
# Wichtig: Andere Mails an externe Ziele verbieten
    reject_unauth_destination,
# Bei eingehenden Mails die Client-IP gegen RBLs checken
    reject_rbl_client zen.spamhaus.org=127.0.0.[2..11],
    reject_rbl_client ix.dnsbl.manitu.net=127.0.0.2
```



```
# Greylisting checken!
    check_policy_service inet:127.0.0.1:10023
# Dynamisch prüfen, ob der Empfänger bei Dovecot/Exchange existiert
    reject_unverified_recipient,
# Was jetzt lebt, darf durch!
    permit
```

Listing 10.6 Die Best Practice der Restrictions

White- und Blacklists pflegen

Die vier access-Dateien haben jeweils einen zweisepaltigen Aufbau: An erster Stelle steht der *Lookup Key*, also der Wert, nach dem gesucht wird. In der zweiten Spalte steht das *Result*, also das Ergebnis. Dabei handelt es sich um eine Aktion, die ausgeführt wird. Die wichtigsten Aktionen sind PERMIT, um eine E-Mail anzunehmen, und REJECT, um die jeweilige E-Mail abzulehnen. Über die Manpage `man 5 access` können Sie jedoch noch zahlreiche weitere Möglichkeiten der access-Map entdecken.

Möchten Sie einen bestimmten Empfänger whitelisten, also Mails an ihn auch dann annehmen, wenn die Client-IP eigentlich durch RBL-Sperrlisten verboten ist, so reicht ein einfacher Eintrag in den `access_recipients`:

```
# Diese Empfängeradresse wird gewhitelistet, empfängt also "immer"
user@example.com      PERMIT
```

Listing 10.7 Empfänger-Whitelisting

Analog dazu können Sie Hostnamen und Absender über `access_helo` oder `access_sender` blocken. Vergessen Sie anschließend das `postmap`-Kommando nicht!

Die Datei `access_client` wird in unserem Beispiel nicht mittels `hash:`, sondern mittels `cidr:` eingebunden, was bedeutet, dass Sie hier auch Subnetzmasken angeben können. Um Mails einer bestimmten Client-IP zu sperren oder Mails einer bestimmten Client-IP immer anzunehmen, können Sie die IP-Adressen wie folgt in die Datei `access_client` eintragen:

```
# Diese einzelne IP-Adresse ist komplett gesperrt
192.168.20.20/32      REJECT

# Und auch dieses ganze /24-Subnetz darf keine Mails einliefern
192.168.100.0/24     REJECT
```

Listing 10.8 Client-Blacklisting

Anders als `hash`-Dateien wird eine `cidr`-Datei von Postfix nicht live abgefragt, sondern fest im Speicher gehalten. Nach Änderungen an der `access_client` müssen Sie darum ausnahmsweise kein `postmap`, sondern ein `postfix reload` ausführen. Für alle Access-Maps in den Restrictions gilt auch hier immer: »*First match wins*.« Wenn Sie möchten, können Sie die Reihenfolge

der access-Map-Prüfungen in den Restrictions anpassen – beispielsweise indem Sie doch zuerst `check_client_access` und dann erst `check_recipient_access` aufrufen. Das obliegt ganz Ihrer Entscheidung für den Fall, dass Sie so bestimmte Anforderungen wie beispielsweise »Ein gewhitelister Client darf trotzdem an gesperrte Empfänger versenden« abbilden möchten.

Nicht existente Domains ablehnen

Stößt Postfix auf einen der Parameter `reject_non_fqdn_*` oder `reject_unknown_*_domain`, prüft er, ob die jeweilige Absender/Empfänger-Domain im DNS überhaupt existiert, das heißt, ob ein MX- oder ein A-Record für diese Domain hinterlegt ist. Ist das bei einem Empfänger nicht der Fall, kann man an diesen auch keine E-Mails senden: Wohin sollte Postfix die Mail auch routen? Und wenn man an einen Empfänger keine Mail senden kann – warum sollte Postfix sie dann überhaupt annehmen? Damit Postfix sie wenige Millisekunden später bouncen und mühsam versuchen müsste, die Mail wieder zurückzusenden?

Gleiches gilt für die inhaltsgleiche Prüfung der Absender-Domain: Wenn die Domain des Absenders gar nicht im DNS steht bzw. keinen MX- und A-Record hat, wird man auf die fragliche E-Mail nicht antworten können. Um eine normale erwünschte E-Mail wird es sich dabei also kaum handeln.

Es ist in aller Regel sinnvoll, zu prüfen, ob die Absender- oder Empfänger-Domain auch für ausgehende E-Mails der eigenen Nutzer existiert. Schließlich wollen Sie weder, dass Ihre eigenen Nutzer mit nicht existenten Domains Mails versenden, geschweige denn macht es für Sie Sinn, Ihren Nutzern Mails an per Definition nicht zustellbare Domains abzunehmen.

In manchen Unternehmen hat es sich jedoch eingebürgert, für interne Hostnamen und Mailadressen `example.local` zu verwenden. Da diese `.local`-Domains nie im DNS auflösbar wären, würden die hier gezeigten Regeln auch den internen Mailverkehr blocken. Nutzen Sie lokal nicht valide Domains, so dürfen Sie diesen Domaincheck erst nach `permit_mynetworks` und/oder `permit_sasl_authenticated` durchführen:

```
smtpd_recipient_restrictions =  
[...]  
# Alle Mails unserer Nutzer/Server erlauben!  
#     permit_sasl_authenticated,  
#     permit_mynetworks,  
# Keine Mails mit nicht existenten Absendern/Empfängern von extern annehmen!  
#     reject_non_fqdn_sender,  
#     reject_non_fqdn_recipient,  
#     reject_unknown_sender_domain,  
#     reject_unknown_recipient_domain,  
[...]
```

Listing 10.9 Domainprüfungen nach »mynetworks«

Interne E-Mails der eigenen User werden so auch mit kaputten Domains akzeptiert; der DNS-Check greift nur noch für Mails von externen Absendern – und diese können und sollen sowieso keine Mails an die `.local`-Adressen schreiben.

RBLs/DNSBLs benutzen

DNS-Blacklists (DNSBLs) gibt es viele im Internet – genauer müsste man jedoch sagen: gab es viele im Internet. Die meisten wurden geschlossen oder werden nicht mehr aktiv gepflegt, weil die Betreiber sich anderweitig orientiert haben oder die Qualität nicht halten konnten. Das heißt nicht, dass RBL-Blacklists schlecht und nicht empfehlenswert seien. Ganz im Gegenteil: Sie können damit mit extrem wenig Ressourceneinsatz bereits rund 60 bis 70% aller eingehenden Spam-Mails blocken. Sie müssen lediglich aufpassen, wem Sie hier vertrauen. Wir empfehlen Ihnen, ausschließlich die unten genannten RBLs zu verwenden, da sie ein seriöses, ständiges Betreiberteam, eine klare Listing-Policy und eine extrem hohe Qualität haben. Erfahrungsgemäß sind Sie damit bestens bedient. Sollten Sie sich weitere RBLs aus anderen Quellen und Anleitungen zusammensuchen wollen, so sollten Sie immer darauf achten, ob die jeweiligen Listen überhaupt noch betrieben werden, und zudem prüfen, nach welchen Bedingungen dort IP-Adressen gelistet werden bzw. auch wieder von der Liste gestrichen werden.

Eine RBL-Abfrage besteht aus einer speziellen DNS-Abfrage an die jeweilige RBL-Domain – in unserem Beispiel `zen.spamhaus.org` oder `ix.dnsbl.manitu.org`. Dabei wird die IP-Adresse des zu überprüfenden Clients (`192.168.10.20`) umgedreht und als Hostname unter der RBL-Domain abgefragt, also als `20.10.168.192.zen.spamhaus.org`. Existiert ein A-Record mit diesem Namen, ist die IP-Adresse auf der Sperrliste enthalten. Liefert das DNS-System für diesen Hostnamen keinen Eintrag, ist die IP-Adresse nicht gelistet. Aus diesem Grund werden RBL-Listen etwas korrekter auch als *DNS-Blacklist* (DNSBL) bezeichnet.

DNSBLs sind sehr effektiv gegen Spam, bergen aber immer die Gefahr, auch erwünschte Mails abzulehnen. RBLs fließen bei jedem Anti-Spam-System als eines von vielen Merkmalen mit in die Bewertung der Spam-Wahrscheinlichkeit einer E-Mail mit ein. Das hat den Vorteil, dass die Mail nicht schon nur deshalb abgelehnt wird, weil der Client auf einer RBL steht. Die Mail muss dann auch sonst noch weitere Merkmale aufweisen, die ein verdächtiges Gesamtbild ergeben. So auch bei Rspamd, das wir Ihnen in Abschnitt 10.3 separat vorstellen.

Wir persönlich ziehen den harten, direkten Einsatz von RBL-Listen im Postfix-Mailserver vor. Die hier erwähnten RBLs sind gut, zuverlässig und listen IP-Adressen nur dann, wenn von dieser IP aus nachweisbar aktuell Spam versandt wird, also »eine akute Gefahr« ausgeht.

Auch wenn das (äußerst geringe) Risiko verbleibt, dass eine IP-Adresse im Einzelfall aufgrund eines Fehlers auf einer RBL landet, überwiegen unseres Erachtens die Vorteile einer schnellen, effektiven, ressourcenschonenden Ablehnung direkt im SMTP-Prozess. Aber das ist am Ende eine Entscheidung, die Sie selbst treffen müssen.

Lizenzbedingungen der Spamhaus-RBL

Spamhaus agiert seit Jahren als weltweite Non-Profit-Organisation gegen Spam und kennt sich in der Szene ausgezeichnet aus. Bitte beachten Sie, dass die Verwendung der Spamhaus-RBL, insbesondere bei mehr als 100.000 eingehenden Mails pro Tag, gegebenenfalls kostenpflichtig sein kann, um die immensen Betriebskosten der über 50 DNS-Server weltweit und des rund um die Uhr aktiven Administrations- und Researcher-Teams wieder refinanzieren zu können. Weitere Informationen finden Sie unter:

<http://www.spamhaus.org/organization/dnsblusage/>

Greylisting

Der größte Teil des heutigen Spams wird über Botnetze versandt, also in aller Regel durch virenfizierte Windows-PCs. Spam von Botnetzen lässt sich hervorragend durch Greylisting abwehren: Dabei werden Verbindungen bislang unbekannter IP-Adressen zunächst temporär mit einem 4xx-Fehler abgelehnt. Da es sich um einen temporären und nicht um einen fatalen Fehler handelt, geht die Mail dabei nicht an den Absender zurück, sondern verbleibt in der Mail-Queue des einliefernden Mailserver und wird wenige Minuten später erneut zugestellt. Da sich der Mailserver beim ersten Versuch in einer kleinen Datenbank die IP-Adresse, die Absender- und die Empfänger-Mailadresse zusammen mit einem Zeitstempel gemerkt hat, wird er einen Wiederholungsversuch als bekannt identifizieren und wird – sofern die übliche Wartezeit von fünf Minuten erreicht wurde – die Mail annehmen und sich die Client-IP in seiner Datenbank für lange Zeit merken, damit zukünftige Mails dieser IP direkt ohne Verzögerung akzeptiert werden. Insofern ist Greylisting in seinen Auswirkungen sehr harmlos: Mailserver werden automatisch gelernt, und bereits bekannte Systeme erfahren gar kein Greylisting mehr. Oder anders ausgedrückt: Nur erste E-Mails bislang unbekannter Systeme werden kurzfristig verzögert.

Durch Greylisting können Sie sehr effektiv prüfen, ob das einliefernde System ein echter normaler Mailserver ist (der queuen kann) oder ob es sich dabei um ein Spam-Botnetz handelt. Denn aus technischen Gründen ist es für Spammer in aller Regel wenig sinnvoll, tatsächlich erneut wiederzukommen: Der Spammer verschwendet durch Wiederholungsversuche wertvolle Ressourcen, die er bei anderen Opfern (im wahrsten Sinne des Wortes) gewinnbringender einsetzen kann.

Denn die Zeit arbeitet gegen den Spammer: Mit jeder Minute, die der Spammer länger am Werk ist, wird die Chance steigen, dass die Virenfektion entdeckt und sein Bot abgeschaltet wird, die IP des Servers geblacklistet oder die von ihm versandte Spam-Mail durch andere Anti-Spam-Verfahren als bekanntermaßen unerwünscht klassifiziert und deswegen abgelehnt wird. Darum gilt für Spammer nach wie vor: »Fire and forget« – schnell rein, schnell raus. Spammer könnten Greylisting durch Wiederholungsversuche überleben, aber sie machen dies größtenteils nicht, weil es mehr Nachteile als Vorteile hat.

Auch unter Virenschutzaspekten ist Greylisting wünschenswert, werden durch Botnetze doch ebenso auch Viren verschickt, und je länger ein Virenausbruch bereits läuft, umso eher wird er von Ihrem Virenkiller erkannt werden können. Insofern ist Greylisting auch ein sehr effektiver und wichtiger Teil eines mehrstufigen Antivirenschutzes und deckt den heiklen Zeitraum zwischen dem ersten Auftreten eines Virus und dem Erscheinen entsprechender Signatur-Updates der Antivirenhersteller ab.

Die beste Greylisting-Implementierung im Zusammenspiel mit Postfix ist *Postgrey*, ein Programm von David Schweikert, der mit *Pflogsumm* ja auch bereits einen schönen Logfile-Reporter für Postfix geschrieben hat. Postgrey ist mit wenigen Handgriffen installiert und bedarf keiner weiteren Pflege. Es nutzt ein paar lokal gespeicherte Berkeley-Datenbanken, in denen es auch selbsttätig aufräumt.



Auch Rspamd führt Greylisting durch

In Abschnitt 10.3 stellen wir Ihnen mit Rspamd noch eine mächtige Anti-Spam/Anti-Virus-Lösung vor. Auch diese implementiert Greylisting als eine von vielen verschiedenen Möglichkeiten, mit Spam umzugehen. Mit Rspamd können Sie genauer und differenzierter steuern, wann und für welche Mails Sie Greylisting einsetzen wollen, z. B. nur dann, wenn sich der Spam-Score der E-Mail in einem verdächtigen Bereich bewegt. Insofern könnten Sie hier auf Postgrey direkt im Postfix-Server verzichten wollen.

Wir persönlich ziehen den Einsatz von Postgrey direkt in Postfix jedoch vor: Es spart Ressourcen und wir haben kein Problem damit, Greylisting auch dann anzuwenden, wenn der Inhalt der E-Mail zunächst unverdächtig aussieht.

Installieren Sie das Paket `postgrey`, und starten Sie den Dämon:

```
mail:~ # systemctl start postgrey
```

Listing 10.10 Der Start von »postgrey«

Ein kurzer Blick in die Prozessliste zeigt, ob Ihre Distribution den Dämon per Default auf einem UNIX-Socket lauschen lässt (so unter CentOS/openSUSE Leap):

```
mail:~ # ps ax | grep postgrey
 9331 ?        Ss      0:00 /usr/sbin/postgrey -d \
                --unix=/var/spool/postfix/postgrey/socket \
                --auto-whitelist-clients
```

Listing 10.11 Aufrufparameter definieren einen UNIX-Socket-Typ.

Falls Ihre Distribution per Default einen TCP/IP-Socket für den Dämon vorbereitet hat (Debian/Ubuntu), sieht der Befehl so aus:

```
mail:~ # ps ax | grep postgrey
23238 ?      Ss    0:00 /usr/sbin/postgrey -d --inet=127.0.0.1:10023 \
        --auto-whitelist-clients
```

Listing 10.12 Aufrufparameter definieren einen TCP-Socket-Typ.

Sie könnten das Verhalten von Postgrey natürlich auch umkonfigurieren, aber am Ende ist es viel einfacher, in Postfix wahlweise einen UNIX- oder einen TCP-Socket zu konfigurieren, als sich mit einer individuellen Postgrey-Konfiguration abzumühen. Postgrey ist ein *Postfix Policy Daemon*, er wird von Postfix über das Postfix-interne *Postfix Policy Delegation Protocol* angesprochen, sodass hier keine Konfigurationsarbeit nötig ist. Sie müssen Postfix in den `smtpd_recipient_restrictions` nur sagen, wo er den Policy-Dämon finden und um Rat fragen soll. Unter openSUSE Leap bzw. bei Verwendung eines UNIX-Sockets lautet der Aufruf:

```
# Greylisting checken!
    check_policy_service unix:postgrey/socket
```

Listing 10.13 Aufruf von »postgrey« durch Postfix über einen UNIX-Socket

Alternativ lautet er unter Debian/Ubuntu bei Verwendung eines TCP-Ports:

```
# Greylisting checken!
    check_policy_service inet:127.0.0.1:10023
```

Listing 10.14 Aufruf von »postgrey« durch Postfix über einen TCP-Socket

Diesen Aufruf müssen Sie an passender Stelle in Ihre `smtpd_recipient_restrictions` integrieren: nach `mynetworks` (damit Ihre eigenen User nicht von Greylisting betroffen sind) und vor der endgültigen Annahme der E-Mail. Listing 10.6 zeigte, wo der beste Platz dafür ist.

Dynamische Empfängervalidierung

Damit Postfix als Relay vor einer anderen Mailserver-Software richtig arbeiten kann, muss Postfix in der Lage sein, unbekannte Empfänger von vornherein ablehnen zu können. Da so gut wie alle Spam- und Virenmails mit gefälschten Absendern unbeteiligter Dritter versendet werden, darf Postfix keinesfalls erst nach der Annahme feststellen, dass eine Mail unzustellbar ist. Würden Mailserver angenommene, aber unzustellbare Spam-Mail verspätet bouncen (*Late Bounce*), würde der unbeteiligte Dritte dermaßen viele Mailrückläufer aus aller Welt erhalten, dass es für ihn einem DDoS (*Distributed Denial of Service*) gleichkäme. Mailserver, die verspätet bouncen, nennt man *Backscatter-Systeme* (*to backscatter* – »zurückwerfen, zurückstreuen«). Backscatter-Systeme werden völlig zu Recht selbst auf RBL-Sperrlisten geführt, da von ihnen eine Gefahr für andere Systeme ausgeht.

Klassischerweise hat man in vorgeschalteten Mailrelays früher statische Empfängerlisten gepflegt oder eine Abfrage eines LDAP- oder AD-Servers eingerichtet. Doch das geht mit Postfix seit vielen Jahren deutlich eleganter, denn Postfix kann über SMTP/LMTP-Verbindungen

zum Zielsystem bequem und vom Nutzer unbemerkt testen, ob dieser eine bestimmte Empfängeradresse akzeptieren würde. Dabei wird keine wirkliche E-Mail abgeschickt, denn nachdem das Zielsystem seine Antwort auf das RCPT TO:-Kommando offenbart hat, sendet Postfix ein QUIT und beendet die Verbindung, ohne eine E-Mail abgesandt zu haben. Das Ergebnis wird für einige Stunden (unbekannter Empfänger) bis hin zu vielen Tagen (bekannter Empfänger) von Postfix gecacht.

Alles in allem ist dies ein ressourcenschonender, verblüffend trivialer und gut funktionierender Mechanismus, der noch nicht einmal besondere Freigaben in der Firewall benötigt. Die sowieso schon vorhandene SMTP/LMTP-Verbindung zum Zielsystem genügt. Selbstverständlich berücksichtigt Postfix dabei für seine Testmails Routing-Tabellen wie die `transport_maps` oder die Umschreibung von E-Mail-Adressen durch die `canonical-` oder `virtual-`Maps. Eine Verify-E-Mail wird darum stets genauso behandelt wie eine echte E-Mail auch. Sie wird halt nur nie abgeschickt.

Per Default ist bei heutigen Postfix-Versionen alles vorbereitet. `address_verify_database` zeigt auf ein Daten-Directory von Postfix, in dem es in einer kleinen Berkeley-Datenbank seinen lokalen Cache der existierenden und nicht existierenden Adressen aufbauen kann:

```
mail:~ # postconf address_verify_map
address_verify_map = btree:$data_directory/verify_cache
```

Listing 10.15 Die »address_verify_map« cacht alle Einträge.

Dieser Cache ist für den Administrator vollständig pflegefrei, denn Postfix räumt seinen `verify_cache` automatisch auf und löscht veraltete Einträge:

```
# Nach 3 Stunden prüft Postfix erneut und erkennt, dass Adressen plötzlich existieren
address_verify_negative_refresh_time = 3h
address_verify_negative_expire_time = 3d

# Nach 7 Tagen prüft Postfix existierende Adressen erneut
address_verify_positive_refresh_time = 7d
address_verify_positive_expire_time = 31d
```

Listing 10.16 Die Caching-Zeiten der »verify«-Datenbank

Existierende Adressen erfahren relativ spät nach sieben Tagen eine erneute Überprüfung. Existiert diese Adresse nicht mehr, wechselt sie nun auf »negativ«, also nicht existent. Ist ein Refresh der Adresse jedoch nicht möglich, zum Beispiel weil das nachgeschaltete System derzeit nicht erreichbar ist, darf Postfix die Adresse bis zu 31 Tage lang als gültig betrachten und Mails annehmen. Auch Mailadressen, von denen Sie nur selten etwas empfangen, sind so mit hoher Wahrscheinlichkeit im Cache und werden von Ihrem Postfix-Relay auch dann noch zuverlässig angenommen, wenn Ihr eigentlich dahintergeschaltetes Mailsystem einige Tage lang nicht erreichbar ist. Nicht existente Adressen erfahren bereits nach drei Stunden

eine erneute Überprüfung, schließlich wollen Sie nicht unnötig lange Mails an frisch eingerichtete Accounts ablehnen. Bereits nach drei Tagen würden nicht existente Adressen jedoch endgültig gelöscht werden, damit Spammer, die viele Adressen ausprobieren, nicht unnötig Ihren Cache verstopfen können. Die Default-Werte von Postfix haben sich hier als sehr praktikabel erwiesen; wir raten Ihnen, diese Werte beizubehalten.

Um die dynamische Empfängervalidierung nutzen zu können, müssen Sie an passender Stelle den Aufruf `reject_unverified_recipient` in Ihre `smtpd_recipient_restrictions` einfügen: nach `mynetworks` und `permit_sasl_authenticated`, damit Sie keinesfalls ausgehende E-Mails bei anderen Providern überprüfen, und auch nach Spam-Schutzmaßnahmen wie RBL- und Greylisting, damit Sie nicht unnötig Verify-Abfragen starten, obwohl Sie die E-Mail des Spammers sowieso nicht annehmen werden. Listing 10.6 zeigte, wo Sie `reject_unverified_recipient` am besten in die Restrictions einbauen.

10.15 Weiterleitungen und Aliasse für Mailadressen

Auf einem Mailserver gibt es häufig auch virtuelle Mailadressen, also Mailadressen, die selbst keinen eigenen Account besitzen, sondern als Weiterleitung auf andere Accounts ausgestaltet sind. Postfix regelt diese Weiterleitungen über die `virtual_alias_maps`-Abfrage, über die Sie beliebige Mailadressen umschreiben, also weiterleiten lassen können. Am einfachsten verwalten Sie Weiterleitungen in der Textdatei `/etc/postfix/virtual`. Definieren Sie dazu in Ihrer `main.cf` den Verweis auf die Datei wie folgt:

```
virtual_alias_maps = hash:/etc/postfix/virtual
```

Listing 10.17 Verweis auf die Virtual-Map von Postfix

Legen Sie dann in `/etc/postfix/virtual` wie folgt Weiterleitungen an:

```
# Alle Mails von Klaus an Susi weiterleiten:
klaus@example.com          susi@example.com

# Den Role-Account "support" an mehrere Admins verteilen lassen
support@example.com        micha@example.com, beate@example.com, joerg@example.com
# Alle anderen Empfänger als catch-all an info@ weiterleiten
@example.com                info@example.com
```

Listing 10.18 Definition von Weiterleitungen in der Virtual-Map

Vergessen Sie nicht, anschließend das Kommando `postmap /etc/postfix/virtual` auszuführen! Ansonsten bekommt Postfix von den Änderungen nichts mit. Übrigens: Haben Sie neben Ihren Domains aus `$relay_domains` noch alternative Domain-Schreibweisen, die jedoch allesamt eine identische Userschaft aufweisen, können Sie mit wenigen Handgriffen diese »Marketing-Domains« annehmen und auf Ihre Haupt-Domain mappen lassen.


```
# Die virtuelle Domain bei Postfix anmelden, damit er Mails an diese Domain annimmt.
# Diese Domains dürfen dann nicht mehr in relay_domains gelistet werden!
example.net          anything

# Alle Mails von example.net auf example.com umschreiben lassen:
@example.net        @example.com
```

Listing 10.19 Definition von virtuellen Domains

Der Userpart der Mailadressen wird dabei stets beibehalten. Aus klaus@example.net wird also klaus@example.com. In Abschnitt 17.16, »Verwaltung von Weiterleitungen für den Mailserver Postfix«, zeigen wir Ihnen übrigens, wie Sie die `virtual_alias_maps`-Abfrage an LDAP anbinden können.



10.1.6 SASL/SMTP-Auth

Ein Mailserver darf keinesfalls als Open Relay Mails für beliebige Nutzer an externe Adressen annehmen und weiterleiten. Spammer würden das mit Handkuss sofort ausnutzen. Lediglich Client-IPs, die aus `$mynetworks` stammen, oder User, die sich mit Usernamen und Passwort authentifizieren (*SMTP-Auth*), dürfen über einen Mailserver an externe Ziele relayen. Doch: Sendet ein Nutzer Username und Passwort, ist Postfix gar nicht in der Lage, diese Angaben selbst zu überprüfen. Stattdessen delegiert er diese Aufgabe über das SASL-Verfahren an IMAP-Server wie Cyrus oder Dovecot, die ja bereits eine funktionierende Userauthentifizierung besitzen müssen.

Während das Zusammenspiel von Postfix und Cyrus traditionell immer recht heikel war, finden Postfix und Dovecot mit wenigen Handgriffen nahtlos zusammen. Im `auth`-Modul von Dovecot wird mit wenigen Zeilen ein eigener Authentifizierungs-socket geöffnet, mit dem sich Postfix verbinden kann und mit dem er von Haus aus umzugehen weiß. Um SASL mit Dovecot in Postfix zu aktivieren, ergänzen Sie die `main.cf` von Postfix wie folgt:

```
smtpd_sasl_auth_enable = yes
smtpd_sasl_path = private/auth
smtpd_sasl_type = dovecot
```

Listing 10.20 Die SASL-Parameter in Postfix

Dies aktiviert nur die SMTP-Auth-Unterstützung in Postfix generell, zieht aber noch keine Konsequenzen in der Mailannahme nach sich. Dazu müssen Sie an passender Stelle in den `smtpd_recipient_restrictions` den Parameter `permit_sasl_authenticated` anführen. Postfix wird dann an dieser Stelle die Mails der Nutzer akzeptieren, die sich zu Beginn des SMTP-Dialogs bereits erfolgreich authentifiziert hatten. Unsere Musterlösung der `smtpd_recipient_restrictions` in Abschnitt 10.1.4, »Die Postfix-Restrictions: Der Schlüssel zu Postfix«, hatte `permit_sasl_authenticated` bereits an passender Stelle vorgesehen (siehe

Listing 10.6). Dort war der Parameter durch eine Raute noch deaktiviert. Aktivieren Sie nun diesen Eintrag:

```
# Mails unserer Nutzer/Server erlauben!  
    permit_sasl_authenticated,  
    permit_mynetworks,
```

Listing 10.21 Die Prüfung der Authentifizierung in den Restrictions

Heutzutage ist es sinnvoll, den eigenen Usern eine Authentifizierung nur geschützt durch SSL/TLS anzubieten. Ergänzen Sie dazu außerhalb der Restrictions Ihre *main.cf* wie folgt:

```
smtpd_tls_auth_only = yes
```

Listing 10.22 Die TLS-Parameter in Postfix

Mit den Anpassungen für Postfix sind Sie nach diesem Schritt fertig, allerdings müssen Sie noch den Authentifizierungs-Socket in Dovecot bereitstellen, also quasi unsere Gegenstelle, die Postfix hier anrufen und um Rat fragen kann. Wenn Dovecot bereits installiert ist, können Sie diese Anpassung schon jetzt vornehmen, ansonsten kehren Sie nach der Installation von Dovecot zu diesem Abschnitt zurück. In Dovecot ist bei neuen Versionen in der Datei */etc/dovecot/conf.d/10-master.cf* bereits ein Authentifizierungs-Socket für Postfix vorbereitet. Suchen Sie die passende Stelle in der *10-master.cf*, prüfen Sie, ob der folgende Eintrag vorbereitet ist, und aktivieren Sie ihn:

```
service auth {  
[...]  
    # Postfix smtp-auth  
    unix_listener /var/spool/postfix/private/auth {  
        mode = 0666  
    }  
[...]  
}
```

Listing 10.23 Der Authentifizierungs-Socket in Dovecot

Nach einem Neustart von Postfix und Dovecot sollte einer Authentifizierung nichts mehr im Wege stehen. Eine erfolgreiche Authentifizierung loggt Postfix eindeutig im Logfile:

```
Jun 29 11:45:49 plasma postfix/smtpd[19664]: 59A19A6BE2: client=unknown[10.0.40.6]:\56828, sasl_method=CRAM-MD5, sasl_username=p.heinlein@heinlein-support.de
```

Listing 10.24 Eine erfolgreiche Authentifizierung im Postfix-Logfile

Übrigens: Wenn Sie Ihren SMTP-Auth-Server und Ihren IMAP-Server getrennt betreiben wollen, müssen Sie trotzdem kurzerhand Dovecot auf dem Postfix-Relay installieren. Sie können Ihre funktionierende Dovecot-Konfiguration von Ihrem IMAP-Server 1:1 kopieren,

damit sichergestellt ist, dass die Abfrage der Authentifizierung funktioniert. Um jetzt jedoch nicht unnötig die eigentlichen Dovecot-Dienste POP3 oder IMAP zu starten, installieren Sie auf Ubuntu lediglich das Paket `dovecot-core`, nicht aber `dovecot-pop3` oder `dovecot-imap`. Unter openSUSE oder CentOS, wo alle Module zusammen paketiert sind, passen Sie in `/etc/dovecot/dovecot.conf` den Eintrag `protocols` wie folgt an, um POP3 und IMAP zu deaktivieren:

```
protocols = none
```

Listing 10.25 Dovecot als reiner Authentifizierungsserver

Dovecot wird nun nur seinen innersten (Authentifizierungs-)Kern starten und klein und harmlos als Partner von Postfix im Hintergrund seinen Dienst verrichten.

10.1.7 SSL/TLS für Postfix einrichten

Nicht nur angesichts des letzten NSA-Skandals ist die SSL/TLS-Verschlüsselung des E-Mail-Verkehrs eine gute Idee. Schon seit über 15 Jahren ist ein konsequenter Schutz von E-Mails und auch von Zugangsdaten per SSL/TLS bei guten Mail Providern eine Selbstverständlichkeit, auch wenn einige Anbieter (bzw. deren Marketing-Abteilung) ihre plötzliche Liebe zu SSL erst in der jüngsten Zeit entdeckt haben.

In Kapitel 31, »Verschlüsselung und Zertifikate«, gehen wir ausführlich auf die Einrichtung einer CA und die Erzeugung eigener Zertifikate ein. Echte Zertifikate mit CA-Unterschrift sind sinnvoll, wenn User mit Desktop-Software auf Ihre Mailserver zugreifen und Sie die Mailübertragung und die Passwortübertragung absichern wollen. Sobald jedoch Mailserver untereinander Mails im Internet übertragen, ist eine SSL-Verschlüsselung sowieso nur optional; manche Provider unterstützen auf Ihren Mailrelays SSL, andere nicht.

Da Mailserver eine E-Mail im Zweifelsfall auch unverschlüsselt an die Gegenseite übertragen würden, kommt es auf die Gültigkeit des verwendeten Zertifikats für sie nicht an. Auch die Verschlüsselung mit nur selbst signierten und nicht überprüften Zertifikaten ist immer noch besser als eine vollständige Klartextübertragung. Für Mailrelays können Sie darum problemlos auf selbst signierte Zertifikate zurückgreifen, und mit diesem charmanten Einzeiler können Sie Key und Zertifikat in einem einzigen Durchgang erzeugen lassen. Dabei wurden früher Key und Zertifikat in zwei getrennten Dateien gespeichert, doch heute ist es besser, Key und Zertifikat(e) in einer gemeinsamen Datei zu speichern – der sog. *Key-Chain*.

```
mail:~ # openssl req -new -newkey rsa:2048 -days 3650 -nodes -x509 -subj \
        /CN=mail.example.com -keyout /etc/postfix/mail.example.com.chain \
        -out /etc/postfix/mail.example.com.chain
mail:~ # chmod 400 /etc/postfix/mail.example.com.chain
```

Listing 10.26 Eine Key-Chain mit selbst signiertem Zertifikat in nur einem Kommando

Haben Sie kommerziell erworbene SSL-Zertifikate, so kopieren Sie diesen Key und seine Zertifikatskette in die Chain-Datei. Achten Sie dabei auf die Reihenfolge: Zuerst steht der Key, dann seine jeweiligen Zertifikate, bevor gegebenenfalls weiterer Key mit seiner Zertifikatskette folgen darf.

Nun müssen Sie noch den Pfad zu Ihrer Key-Chain in der *main.cf* von Postfix ergänzen:

```
# Ausgehend SSL/TLS aktivieren
smtp_tls_security_level = dane
smtp_dns_support_level = dnssec

# Eingehend SSL/TLS aktivieren
smtpd_tls_security_level = may
smtpd_tls_cert_file = /etc/postfix/mail.example.com.chain
# Alte Postfix-Versionen und Anleitungen sehen noch getrennte Dateien
# für Key und Zertifikate vor:
# smtpd_tls_cert_file = /etc/ssl/certs/mail.example.com.crt
# smtpd_tls_key_file = /etc/ssl/private/mail.example.com.key
# smtpd_tls_CAfile = /etc/ssl/certs/ca-zwischenzertifikat.crt
```

Listing 10.27 SSL/TLS in Postfix aktivieren

In der jüngeren Vergangenheit ist *Perfect Forward Secrecy* (PFS) in den Mittelpunkt der Aufmerksamkeit geraten. Mittels spezieller DHE-Ciphers kann die SSL-Kommunikation besonders gegen eine nachträgliche Entschlüsselung abgesichert werden. Optimieren Sie Ihr SSL-Setup, indem Sie über das OpenSSL-Kommando besondere Diffie-Hellman-Files erzeugen:

```
mail:~ # openssl gendh -out /etc/postfix/dh_2048.pem -2 2048
```

Listing 10.28 Erzeugung der notwendigen Diffie-Hellman-Files

Gegen das Diffie-Hellman-Verfahren existieren Angriffe, die die Schlüssellänge schwächen. Aus diesem Grund sollten keine DH-Schlüssel mehr mit 1024 Bit Länge, sondern gleich mit 2048 Bit Länge erzeugt werden. Diese werden durch die Angriffsmöglichkeit zwar auch etwas geschwächt, sind damit aber immer noch dermaßen stark, dass sie als sicher gelten.

Postfix kennt keinen eigenen DH-Parameter für 2048 Bit Schlüssellänge. Stattdessen können Sie über den vorhandenen Parameter `smtpd_tls_dh1024_param_file` direkt auf ein File mit 2048 Bit verweisen, und Postfix wird dann 2048er-Schlüssel verwenden. Lassen Sie sich in diesem Fall also von dem Parameternamen nicht verwirren.³

Tragen Sie in der *main.cf* von Postfix also Folgendes ein:

3 Genau genommen könnten Sie hier auch noch größere DH-Schlüssellängen verwenden, doch bringt dies nach heutigem Kenntnisstand, gemessen an der Mehrbelastung des Systems, keine sinnvollen Vorteile mehr.

```
# 2048-Bit-DH-Schlüssel zuweisen
smtpd_tls_dh1024_param_file = /etc/postfix/dh_2048.pem
```

Listing 10.29 »Perfect Forward Secrecy« in Postfix aktivieren

Übrigens: Auf <https://de.ssl-tools.net/mailservers> können Sie Ihren eigenen Mailserver bequem durchchecken lassen.

10.2 POP3/IMAP-Server mit Dovecot

Dovecot hat sich vom späten Newcomer zum technologisch führenden IMAP-Server entwickelt, auf dem fast alle IMAP-Systeme heutzutage laufen. Mächtig, robust, übersichtlich, logisch: Dovecot ist des Admins Liebling. Mit wenigen Handgriffen richten Sie sich mit Dovecot einen POP3- und IMAP-Server für Familie, Arbeitsgruppen, Unternehmen oder auch für kleine ISPs mit fünfstelligen Nutzerzahlen ein.⁴

10

10.2.1 Installation der Dovecot-Pakete

Seit einigen Jahren ist Dovecot im Versionszweig 2.3.x erschienen, die Reihe 2.2.x sollte nicht mehr verwendet werden. Allerdings sind die Pakete in den Distributionen oft etwas älter, und gerade die dort enthaltenen sehr »frühen« ersten Pakete der damals neuen 2.3er-Reihe haben einige Bugs und Probleme, die durch den großen Versionssprung entstanden sind.

► Debian, Ubuntu, CentOS

Für Debian, Ubuntu und CentOS installieren Sie darum am besten die jeweils letzten Releases aus dem offiziellen Dovecot-Repository. Auf <https://repo.dovecot.org> finden Sie kurze Anleitungen, wie Sie das Repository in den verschiedenen Distributionen hinzufügen. Achten Sie bei diesen Distributionen darauf, dass die Dovecot-Pakete ähnlich wie bei Postfix sehr kleinteilig sind. Verschaffen Sie sich einen Überblick mittels `apt-cache search dovecot` bzw. `dnf search`, und installieren Sie dann:

- das Hauptpaket `dovecot-core` und die weiteren Module,
- `dovecot-lmtp`,
- `dovecot-imapd`,
- `dovecot-pop3d`,
- `dovecot-managesieved` und
- `dovecot-sieve`.

⁴ Sie können mit Dovecot auch ISPs mit sieben- oder achtstelligen Nutzerzahlen betreiben; dann liegen der Konfiguration allerdings doch erheblich andere Konzepte zugrunde. Das Dovecot-Buch von Peer Heinlein, *Dovecot: POP3/IMAP-Server für Unternehmen und ISPs*, <https://www.dovecot-buch.de>, beleuchtet das in allen Facetten.

► openSUSE

Für openSUSE sind aktuelle und gut gepflegte Pakete vorhanden, sodass Sie die normalen Versionen aus der Distribution verwenden können. Sollten Sie irgendwann doch noch neuere Pakete verwenden wollen, so empfehlen wir Ihnen eine Paketsuche auf <https://software.opensuse.org>. Dort werden Sie schnell unter VERSUCHSPAKETE auf das spezielle Repository `server:mail` stoßen, das stets gut gepflegt ist.

Unter openSUSE sind alle wichtigen Dovecot-Module bequem in einem Paket zusammengefasst. Installieren Sie lediglich das Paket `dovecot`. Nur wenn Sie später mit MySQL- oder PostgreSQL-Unterstützung weitermachen wollen, benötigen Sie auch die Pakete `dovecot-backend-mysql` oder `dovecot-backend-pgsql`.

10.2.2 Vorbereitungen im Linux-System

Es empfiehlt sich, alle Dovecot-Accounts unter einer einheitlichen User- und Gruppenkennung zu betreiben, um unabhängig vom System zu sein und um Zugriffsprobleme zu vermeiden, sollten Sie später fortgeschrittene Konfigurationen mit Shared Foldern zwischen den Nutzern betreiben wollen. Zudem würden getrennte Userkennungen hier sowieso keine nennenswerten weiteren Sicherheitsvorteile bringen, da Dovecot ohnehin auf die Maildaten aller Nutzer zugreifen können muss. Der User- und Gruppenname `vmail` und die User- und die Gruppen-ID `10000` haben sich als Best Practice für die Speicherung der Dovecot-Mails und als gemeinsame System-UID aller Dovecot-Nutzer eingebürgert:

```
mail:~ # groupadd -g 10000 vmail
mail:~ # useradd -u 10000 -g 10000 -s /bin/false vmail
```

Listing 10.30 Anlegen des Users »vmail«

Außerdem können Sie bei dieser Gelegenheit gleich den Ordner vorbereiten, in dem Dovecot später seine E-Mails speichern soll. Je nach persönlicher Vorliebe bietet sich hier `/srv/vmail` oder `/var/vmail` an – nicht zu verwechseln mit `/var/mail`, dem »klassischen« Pfad des Linux-Systems für die Mails der echten Systemuser:

```
mail:~ # mkdir /srv/vmail
mail:~ # chown vmail:vmail /srv/vmail
mail:~ # chmod 770 /srv/vmail
```

Listing 10.31 Anlegen des richtigen Verzeichnisses zur Speicherung der E-Mails

10.2.3 Log-Meldungen und Debugging

Die Konfiguration von Dovecot spielt sich in `/etc/dovecot/dovecot.conf` sowie in zahlreichen aufgesplitteten Unterdateien in `/etc/dovecot/conf.d/*` ab. Dabei ist es eigentlich egal, wo ein Parameter tatsächlich eingetragen wird. Aus Sicht von Dovecot sind alle Konfigurations-

dateien ein großes Ganzes. Aber um als Administrator den Überblick zu behalten und um die vorhandene Dokumentation in den Kommentaren der Konfigurationsdatei besser im Blick zu haben, empfiehlt es sich, die jeweiligen Parameter stets an Ort und Stelle zu pflegen, wo sie bereits vorbereitet und kommentiert sind.

Bevor Sie mit Dovecot starten, sollten Sie zuerst die Gesprächigkeit von Dovecot in seinen Logfiles erhöhen. Die ausführlicheren Meldungen können am Anfang sehr hilfreich sein, um zu verstehen, was Dovecot denkt und wie es vorgeht; später im Produktivbetrieb sollten Sie insbesondere die Debug-Parameter wieder zurücknehmen. In `/etc/dovecot/dovecot.conf` sollten Sie zuerst Folgendes festlegen:

- ▶ `verbose_proctitle = yes`

Ist diese Einstellung aktiv, blendet Dovecot in der Prozessliste bei seinen `pop3d`- und `imapd`-Prozessen ein, welcher User von welcher IP diesen Prozess initiiert hat; gegebenenfalls wird auch ergänzt, ob SSL aktiv ist:

```
1770 ? S 0:00 dovecot/imap [klaus@example.com 10.0.40.6 IDLE, TLS]
```

Listing 10.32 Dovecot zeigt übersichtlich, wem ein Prozess gehört.

Diese Einstellung ist sinnvoll, um das System zu überwachen. Ressourcenfresser können so leicht gefunden und identifiziert werden. Aus Datenschutzgründen dürfen Sie diese Einstellung jedoch nicht auf Systemen vornehmen, auf denen neben `root` auch weiterhin normale User Shell-Logins haben und die Prozessliste sehen könnten.

Des Weiteren können Sie in `/etc/dovecot/conf.d/10-logging.conf` Folgendes anpassen:

- ▶ `auth_verbose = yes`
führt zu umfangreichen Log-Meldungen rund um den Authentifizierungsprozess.
- ▶ `auth_debug = yes`
debuggt den Authentifizierungsprozess in jedem Detail, schlüsselt auch die LDAP- oder Datenbankkommunikation auf (sinnvoll zu Beginn, wenig sinnvoll im Live-Betrieb).
- ▶ `mail_debug = yes`
debuggt ausführlich, wie Dovecot das Mailverzeichnis des Users sucht und das verwendete Speicherformat erkennt.

10.2.4 User-Authentifizierung

Dovecot ist gleichzeitig ein POP3- und IMAP-Server und bietet verschiedene Speicherformate an, um die Maildaten der Nutzer sicher auf der Festplatte zu verwahren. Für IMAP-Server mit bis zu 3 oder 4 TByte Speichervolumen sollten Sie dabei auf das altbewährte Maildir-Format zurückgreifen, bei dem Sie als Administrator nicht viel falsch machen können und bei dessen Administration wenig zu beachten ist. Tragen Sie darum in die `/etc/dovecot/conf.d/10-mail.conf` Folgendes ein:

```
mail_location = maildir:~/Maildir
```

Listing 10.33 Festlegung von Maildir als Standard-Speicherformat

Ihr System wäre damit bereits fertig benutzbar für alle User, die Sie in */etc/passwd* angelegt haben. Doch für einen Mailserver ist es meistens erstrebenswerter, System- und Mailnutzer nicht miteinander zu vermischen. Dovecot kennt darum die Möglichkeit, in */etc/dovecot/users* seine ganz eigene Liste mit Usernamen und Passwörtern zu pflegen. Die Syntax von */etc/dovecot/users* ist dabei identisch mit */etc/passwd*, sieht man davon ab, dass im zweiten Feld auch gleich das Passwort mit gespeichert wird:⁵

```
mail:~ # cat /etc/dovecot/users
# Syntax: username:passwort:uid:gid:home:realname:extra_fields
bla@example.org:{PLAIN}test:10000:10000::/srv/vmail/example.org/bla::
blubb@example.com:{PLAIN}test:10000:10000::/srv/vmail/example.com/blubb::
```

Listing 10.34 »/etc/dovecot/users« entspricht in Aufbau und Syntax der »/etc/passwd«.

Anders als in der »echten« */etc/passwd* können Sie in */etc/dovecot/users* nicht nur Usernamen, sondern gleich echte Mailadressen als Login-Kennung angeben. Diese sind stets eindeutig und haben zudem den Vorteil, dass Sie Mails nicht in Postfix von der Mailadresse auf die Login-Kennung umschreiben müssen. Stattdessen können Sie – wie oben gezeigt – die hier angelegten Mailadressen direkt über die Postfix-Tabelle *relay_domains* bis ins Ziel routen. Auch wenn Sie später vielleicht die serverseitige Mailfilterung mit *Sieve* einrichten wollen, um Ihre User darüber E-Mails sortieren oder Autoresponder einrichten zu lassen, ist es hilfreich, wenn E-Mails bis zum Schluss an die Kennung adressiert sind, die Sie auch im Mailheader als Mailadresse finden.

Nachdem Sie Dovecot neu gestartet haben, können Sie über das *doveadm*-Kommando prüfen, ob die User-Authentifizierung richtig funktioniert:

```
mail:~ # doveadm user blubb@example.com
field  value
uid    10000
gid    10000
home   /srv/vmail/example.com/blubb
mail   maildir:~/Maildir

mail:~ # doveadm auth test blubb@example.com
Password:
passdb: blubb auth succeeded
```

Listing 10.35 »doveadm« debuggt, ob und wie ein Nutzer vorhanden ist.

5 Wie das vor Einführung von */etc/shadow* in der */etc/passwd* früher auch der Fall war.

In älteren Dovecot-Versionen hieß das Kommando nicht `doveadm auth test`, sondern lediglich `doveadm auth`:



```
mail:~ # doveadm auth blubb@example.com
Password:
passdb: peer auth succeeded
```

Listing 10.36 »`doveadm auth`« gilt bei alten Dovecot-Versionen.

In Abschnitt 17.17, »Benutzerauthentifizierung von Dovecot über LDAP«, zeigen wir Ihnen ausführlich, wie Sie die User-Authentifizierung von Dovecot an LDAP anbinden können.



10.2.5 Aktivierung des LMTP-Servers von Dovecot

10

Am besten ist es, wenn eingehende E-Mails per LMTP an Dovecot gesandt werden, damit dieser sie in die Postfächer der Nutzer speichern kann. Dovecot bringt dafür von Haus aus einen aktivierten LMTP-Server mit, der in `/etc/dovecot/conf.d/10-master.cf` als UNIX-Socket vorbereitet ist. Postfix könnte Mails an diesen UNIX-Socket zustellen, doch müsste man dann im LMTP-Modul von Postfix auf eine chroot-Konfiguration verzichten. Auf einen außerhalb seiner chroot-Umgebung liegenden UNIX-Socket kann Postfix nicht zugreifen. Besser ist es darum, Dovecot auf `localhost` den LMTP-Port 24 öffnen zu lassen, den Postfix auch aus seiner chroot-Umgebung heraus erreichen kann. Suchen Sie dafür die Definition des LMTP-Servers in der `10-master.cf`, und aktivieren Sie zusätzlich den `inet_listener` wie folgt:

```
service lmtp {
    unix_listener lmtp {
        #mode = 0666
    }

    # Create inet listener only if you can't use the above UNIX socket
    inet_listener lmtp {
        # Avoid making LMTP visible for the entire internet
        address = 127.0.0.1
        port = 24
    }
}
```

Listing 10.37 Aktivierung des LMTP-Sockets auf TCP-Port 24

Nach einem Restart von Dovecot sollte der LMTP-Port empfangsbereit sein; falls nicht, finden sich sicherlich Fehlermeldungen im Logfile:

```
mail:~ # telnet localhost 24
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
```

```
220 mail.heinlein-support.de Dovecot ready.
quit
221 2.0.0 OK
Connection closed by foreign host.
```

Listing 10.38 Test-Login per LMTP bei Dovecot

Im Postfix-Teil (siehe Abschnitt 10.1.3, »Postfix als Relay vor Exchange, Dovecot oder anderen Backends«) hatten wir in Listing 10.3 das Mail-Routing in */etc/postfix/relay_domains* bereits auf Basis von LMTP an einen lokalen LMTP-Server vorbereitet. Zur Erinnerung:

```
mail:~ # cat /etc/postfix/relay_domains
# Diese Domain wird an einen lokalen Dovecot/Cyrus-Server weitergegeben
example.com    lmtp:[127.0.0.1]
```

Listing 10.39 Die Syntax der kombinierten Relay-Domains- und Routing-Tabelle

10.2.6 Einrichten von SSL/TLS-Verschlüsselung

Vermutlich haben Sie bereits SSL-Zertifikate für Postfix erzeugt oder eingebunden, wie in Abschnitt 10.1.7, »SSL/TLS für Postfix einrichten«, gezeigt. Die dort erzeugten Zertifikate können Sie auch direkt für Dovecot weiterverwenden, sofern die Zertifikatshostnamen zu den Hostnamen passen, unter denen Ihre User Ihren Mailserver ansprechen. Passen Sie dazu */etc/dovecot/conf.d/10-ssl.conf* wie folgt an:

```
# Mögliche Werte: yes | no | required
ssl = required
ssl_cert = </etc/ssl/certs/mail.example.com.crt
ssl_key = </etc/ssl/private/mail.example.com.key
```

Listing 10.40 Definition von Key und Zertifikaten für Dovecot

Setzen Sie `ssl=yes`, so wird SSL zwar angeboten, Clients könnten sich aber weiterhin unverschlüsselt mit dem Server verbinden. Setzen Sie also lieber wie gezeigt `ssl = required`. In heutigen Zeiten sollte eine durchgängige Verschlüsselung erstrebenswert bis selbstverständlich sein.



Die führende spitze Klammer vor den beiden Pfadangaben zu Key und Zertifikat ist hier ausnahmsweise richtig. Sie dient dazu, dass Dovecot die Keys auf eine besondere Art und Weise einliest, damit sie nicht in falsche Hände geraten.

Wenn Ihre CA Zwischenzertifikate verwendet, dann tragen Sie diese ebenfalls mit in die Datei */etc/ssl/certs/mail.example.com.crt* ein. Beginnen Sie dabei erst mit dem Zertifikat Ihres Servers, und fahren Sie dann jeweils mit dem nächsthöheren Zwischenzertifikat fort. Analog zu Postfix sind auch hier Diffie-Hellmann-Schlüssel wünschenswert. Setzen Sie in */etc/dovecot/conf.d/10-ssl.conf* folgenden Eintrag:

```
# DH file to use.
ssl_dh=</etc/dovecot/dh.pem
```

Listing 10.41 Diffie-Hellman-Schlüssel aktivieren

Nachdem Sie Ihren Nutzern SSL/TLS zur Verfügung gestellt haben, sollten Sie verhindern, dass diese sich über die Passwortverfahren PLAIN und LOGIN im Klartext, und damit für Dritte mitlesbar, am Server anmelden können. Stattdessen sollten Klartextanmeldungen bei sicheren Verbindungen zulässig sein – also wenn SSL/TLS verwendet wird oder wenn die Verbindung von localhost aus erfolgt. Prüfen Sie dazu den entsprechenden Eintrag zu Beginn der */etc/dovecot/conf.d/10-auth.conf*:

```
disable_plaintext_auth = yes
```

Listing 10.42 Keine ungesicherten Passwortübertragungen zulassen

10

10.2.7 Der Ernstfall: Der IMAP-Server erwacht zum Leben

Sie haben jetzt

- ▶ ein Postfix-Relay,
 - das Empfänger über eine dynamische Adressvalidierung bei Dovecot auf Existenz prüft,
 - Spam und Viren mittels Greylisting oder RBL-Checks filtert
 - und am Ende Mails für Domains aus */etc/postfix/relay_domains* annimmt
 - sowie diese Mails per LMTP an Dovecot ausliefert.
- ▶ einen POP3/IMAP-Server mit Dovecot,
 - der seine Nutzer bequem in */etc/dovecot/users* vorhält,
 - Mails an diese Nutzer per LMTP von Postfix annimmt
 - und diese Mails im Maildir-Format in die Home-Mailverzeichnisse der (virtuellen) Nutzer speichert
 - sowie diese Mails an eingeloggte Nutzer natürlich auch wieder herausgibt.

Nun ist es Zeit für einen Testlauf. Nehmen Sie sich einen Mail-Client Ihrer Wahl (zum Beispiel Outlook, Thunderbird oder Evolution), hinterlegen Sie dort die richtigen Zugangsdaten, und prüfen Sie, ob sich der Mail-Client grundsätzlich über IMAP mit Dovecot verbinden kann. Falls nicht, finden Sie Aufschlussreiches im Mail-Logfile. Klappt der IMAP-Login, können Sie per SMTP eine Mail an Ihren Testnutzer bei Postfix einliefern lassen und den Weg durch Postfix, Amavis und Dovecot verfolgen:

```
Jul 27 19:22:22 mail postfix/smtpd[11357]: ED38B6DBE: client=localhost [127.0.0.1]
Jul 27 19:22:22 mail postfix/cleanup[11358]: ED38B6DBE: message-id=\
<20200727171243.ED38B6DBE@mail.heinlein-support.de>
```

```
Jul 27 19:22:22 mail postfix/qmgr[11325]: ED38B6DBE: from=<p.heinlein@mailbox.org>,\
size=1128, nrcpt=1 (queue active)
Jul 27 19:22:22 mail postfix/lmtp[11359]: ED38B6DBE: to=<k.test@example.com>,\
relay=127.0.0.1[127.0.0.1]:24, delay=31, delays=31/0/0/0.01, \
dsn=2.0.0, status=sent (250 2.0.0 <k.test@example.com> \
3x4FOCoz1VOULAAAluV1ow Saved)
Jul 27 19:22:22 mail postfix/qmgr[11325]: ED38B6DBE: removed
Jul 27 19:22:22 mail dovecot: lmtp(11412, k.test@example.com): 3x4FOCoz1VOULAAAlu\
V1: msgid=<20140727171243.ED38B6DBE@mail.heinlein-support.de>:\
saved mail to INBOX
```

Listing 10.43 Die Spur einer E-Mail im Mail-Logfile (gekürzt)

Bei dieser E-Mail ist alles gut gegangen. Doch die folgenden Beispiele zeigen, wie sich die Ablehnung einer E-Mail beispielsweise in den Log-Meldungen widerspiegelt hätte:

- ▶ Eine wegen Greylisting temporär (!) abgelehnte E-Mail:

```
Jul 27 17:50:00 mail postfix/smtpd[6418]: NOQUEUE: reject: RCPT from \
unknown[67.238.xxx.xxx]: 450 4.2.0 <user@example.com>: \
Recipient address rejected: Greylisted, see \
http://postgrey.schweikert.ch/help/example.com.html; from=<info@example.net> \
to=<user@example.com> proto=SMTP helo=<dummy.example.net>
```

Listing 10.44 Das Greylisting verweigert temporär die Annahme.

- ▶ Ein durch eine RBL geblacklisteter Client:

```
Jul 27 19:21:46 mail postfix/smtpd[19374]: NOQUEUE: reject: RCPT from \
v113-52-51-118.myvps.vn[113.52.51.118]: 554 5.7.1 Service unavailable; \
Client host [113.52.51.118] blocked using zen.spamhaus.org \
http://www.spamhaus.org/query/bl?ip=113.52.51.118 / \
http://www.spamhaus.org/sbl/query/SBL188125; from=<info@example.net> \
to=<user@example.com> proto=ESMTP helo=<dummy.example.net>
```

Listing 10.45 Der absendende Server steht auf einer Blacklist.

10.2.8 Dovecot im Replikations-Cluster

Nachdem Sie nun einen einzelnen Rechner mit Postfix und Dovecot zum voll funktionsfähigen SMTP- und IMAP-Server aufgebaut haben, lohnt noch der Seitenblick in Richtung »Hochverfügbarkeit« und indirekt auch in Richtung »Backup«. Dovecot besitzt die geniale Fähigkeit, zwei ansonsten identisch aufgebaute Dovecot-Server zu einem bidirektionalen Replikations-Cluster zusammenzuschalten.

Dabei heißt *Replikations-Cluster*: Jede Änderung auf der einen Seite wird über ein internes Dovecot-Protokoll zur anderen Seite repliziert.

Dabei speichert jede Seite ihren Mailbestand autark individuell auf einer eigenen Festplatte. Anders als bei Replikationsdateisystemen wie DRBD werden dabei Schäden am Dateisystem nicht ebenfalls stoisch zum Clusterpartner übertragen, da Änderungen nur als Mail-Event von Dovecot an den Clusterpartner übertragen werden. Wird durch den Administrator auf einer Seite versehentlich ein ganzer Dateibaum mit Mails gelöscht, gleichen sich beide Systeme beim nächsten Replikationslauf wieder ab und restaurieren den »illegal« verschwundenen Mailbestand wieder. Und *bidirektional* heißt: Dabei können auf jeder Seite Änderungen stattfinden, sodass es gar nicht notwendig ist, einen Knoten als Master und den anderen als Slave zu definieren.

Stattdessen können Sie den Dovecot-Cluster ganz bequem als Master-Master-Replikation betreiben. Damit entfallen auch typische HA-Probleme wie *Split Brain* oder *STONITH*, bei denen durch verschiedene Techniken dafür gesorgt werden muss, dass zur gleichen Zeit jeweils nur ein Knoten aktiv ist. Bei Dovecot können Nutzer zeitgleich auf jedem Knoten aktiv sein und damit auch über ein einfaches Loadbalancing-Round-Robin verteilt werden. »Können« heißt aber eben nicht »müssen«, sodass wir Ihnen empfehlen möchten, zwar ein Master-Master-Setup aufzubauen, durch eine einfache Loadbalancer- oder HA-Konfiguration aber dafür zu sorgen, dass die User trotzdem nur jeweils einem Knoten zugeteilt werden. Sie behalten so besser den Überblick, haben alle Logfiles zentral und können ihr Mailsystem leichter administrieren. Doch dieses Balancing geschieht nun außerhalb von Dovecot, der sich trotzdem weiterhin in einer Master-Master-Replikation sieht und sehen soll.⁶

10.2.9 Einrichtung der Replikation

Um eine Replikation zwischen zwei Dovecot-Knoten zu initiieren, müssen beide einen TCP-Port für das Dovecot-eigene *doveadm*-Protokoll öffnen und einen auf beiden Seiten identisch eingestellten symmetrischen Schlüssel verwenden. Des Weiteren müssen einige Plug-ins und Services aktiviert werden, die die stattfindenden Mail-Events überwachen und an die jeweils andere Seite melden. Nachdem Sie entsprechend diesem Kapitel einen zweiten, identischen Mailserver aufgebaut haben (oder bei einer virtuellen Maschine die vorhandene Installation geklont haben), können Sie auf beiden Servern die Konfigurationsdatei */etc/dovecot/conf.d/17-replication.conf* einspielen:

```
mail_plugins = $mail_plugins notify replication

service aggregator {
    fifo_listener replication-notify-fifo {
```

⁶ Für größere Setups kennt Dovecot dazu weitere Komponenten wie den *Dovecot Director*, der für Caching-Vorteile sorgt, indem sich mehrere Logins des gleichen Users vorzugsweise auf demselben Dovecot-Knoten abwickeln lassen; diese Optimierungen sind für fünf-, sechs- oder siebenstellige Userzahlen interessant und können im kleineren Bereich ignoriert werden. Aus diesem Grund gehen wir auf den Dovecot Director hier nicht weiter ein.

```
        user = vmail
    }

    unix_listener replication-notify {
        user = vmail
    }
}

service replicator {
    process_min_avail = 1
    unix_listener replicator-doveadm {
        mode = 0600
        user = vmail
    }
}

service doveadm {
    inet_listener {
        port = 12345
    }
}

doveadm_port = 12345
doveadm_password = secretpassword
replication_dsynchronisation_parameters = "-d -n INBOX -l 30 -U"

plugin {
# mail01 repliziert zu mail02 alias .1
# Auf mail01 also aktivieren, auf mail02 deaktivieren:
    mail_replica = tcp:192.168.10.1
# mail02 repliziert zu mail01 alias .2
# Auf mail02 also aktivieren, auf mail01 deaktivieren:
#     mail_replica = tcp:192.168.10.2
}
```

Listing 10.46 Konfiguration der Replikation auf beiden Mailservern

Passen Sie dabei die in der Beispielkonfiguration genannten IP-Adressen entsprechend an. Wenn Sie Dovecot neu gestartet haben und damit auch das oben genannte notify-Plug-in aktiviert haben, wird nun auch Ihr doveadm-Tool die replication-Kommandos unterstützen:

```
root@mail:~# doveadm replicator
usage: doveadm [-Dv] [-f <formatter>] replicator <command> [<args>]
    add          [-a <replicator socket path>] <user mask>
```

```

dsync-status [-a <replicator socket path>]
remove      [-a <replicator socket path>] <username>
replicate   [-a <replicator socket path>] [-f] [-p <priority>] <user mask>
status      [-a <replicator socket path>] [<user mask>]

```

Listing 10.47 »doveadm«-Befehlsübersicht für die Replikation

Ein `doveadm replicator replicate '*'` startet einen Replikationsabgleich über alle User. Kurz nach dem Kommando sollten Sie auf dem zweiten Server hektische Betriebsamkeit im Logfile sehen und auch im Dateisystem beobachten können, wie »von Zauberhand« der Mailspeicherbereich des ersten Servers auf den zweiten Server übertragen wird. Wenn Sie sich nun per POP3 oder IMAP auf dem zweiten Server einloggen, finden Sie dort das komplette Postfach vor, und gleichzeitig wird jede dort vorgenommene Änderung ebenfalls auf den ersten Server zurückübertragen – eben bidirektionale Replikation.

Dass Sie über `doveadm replicator remove` einzelne User von der Replikation ausnehmen möchten, wird eher selten der Fall sein. Ganz analog brauchen Sie dann auch diese gesperrten User nie über `add` hinzuzufügen – per Default repliziert Dovecot von Beginn an jeden Nutzer. Praxisrelevanter ist hier eher das Kommando `doveadm replicator dsync-status`, mit dem Sie sich einen Überblick verschaffen können, womit sich die Replikation derzeit »live« beschäftigt. Per Default werden maximal zehn Replikationsaktivitäten parallel ausgeführt. Da aber nach der allerersten Replikation nur noch Änderungen übertragen werden, ist es nicht ungewöhnlich, dass kaum oder keine konkrete Aktivität zu sehen ist:

```

root@host:~# doveadm replicator dsync-status
username                               type      status
user8@example.org                      incremental Waiting for dsync to finish
-                                       -        Not connected
-                                       -        Not connected
user1@example.org                      incremental Waiting for dsync to finish
user3@example.org                      incremental Waiting for dsync to finish
-                                       -        Not connected
user6@example.org                      incremental Waiting for dsync to finish
-                                       -        Not connected
-                                       -        Not connected
-                                       -        Not connected

```

Listing 10.48 Live-Statusübersicht der Replikation

Über das Kommando `doveadm replicator status` hingegen können Sie sich anzeigen lassen, wie der generelle Systemzustand ist und wie viele Einträge derzeit gegebenenfalls noch in der Replikationswarteschlange auf Abarbeitung warten:

```

root@host:~# doveadm replicator status
Queued 'sync' requests      0

```

```
Queued 'high' requests      0
Queued 'low' requests       3
Queued 'failed' requests    0
Queued 'full resync' requests 7016
Waiting 'failed' requests    0
Total number of known users 45341
```

Listing 10.49 Statusübersicht der Replikation

Dabei dürfen Sie sich auch im laufenden Betrieb nicht wundern, wenn immer wieder User auf einen *full resync*-Request warten, da Dovecot per Default jeden User alle 24 Stunden einem *full resync* unterzieht, um gegebenenfalls verlorene einzelne Events aus dem inkrementellen Abgleich abzufangen. Hinweise auf die Aktivitäten der Replikation und gegebenenfalls auch Hintergründe zu etwaigen Fehlern finden Sie wie gewohnt im Mailserver-Logfile.

Nun haben Sie zwei sich gegenseitig absichernde IMAP-Cluster, die durch ihre jeweils autonome Datenspeicherung auch für ein gewisses Grund-Backup sorgen. Dieses schützt Sie zwar nicht davor, dass ein User versehentlich alle seine Mails löscht, wohl aber vor einem Hardware-schaden oder gegebenenfalls auch einfach vor »menschlichem Versagen« an der Tastatur, wenn Sie Dateien auf dem Server löschen.

10.2.10 Hochverfügbare Service-IP

Damit der Cluster gemeinsam und ausfallsicher für den Nutzer seine Dienste anbinden kann, benötigen Sie nun noch eine gemeinsame Service-IP (Cluster-IP), die wahlweise auf dem einen oder dem anderen Server liegt. Das zu realisieren, ist auf vielen verschiedenen Wegen möglich, beispielsweise über Heartbeat/Pacemaker, wie es in Kapitel 19 beschrieben wird. Sie benötigen dabei nur eine absolute Minimalkonfiguration (wie in Abschnitt 19.6.1 gezeigt), die die besagte Service-IP mal auf dem einen Server, mal auf dem anderen Server hochfährt. Dovecot und alle anderen Dienste laufen auf den Servern weiterhin nonstop durch und müssen darum auch nicht über Heartbeat/Pacemaker gemanagt werden.

Wenn Sie jedoch nicht bereits mit Heartbeat/Pacemaker aus anderen Projekten vertraut sind, ist die für Sie sicher schlankere und übersichtlichere Lösung der Einsatz von VRRP (alias *keepalived*). Dazu wird auf beiden Servern ein *keepalived*-Dämon installiert, sodass sich beide über IP-Broadcasts gegenseitig überwachen und koordinieren. Auf dem jeweils zum Master gekürten System wird die eingestellte Service-IP aktiv geschaltet. Installieren Sie dafür *keepalived*, und passen Sie */etc/keepalived/keepalived.conf* an.

Dabei benötigen Sie auch hier nur eine absolute Minimalkonfiguration, denn auch hier ist es nur notwendig, die Service-IP, nicht aber den eigentlichen Dovecot-Dienst zu verwalten, da ja beide Dovecot-Knoten trotzdem noch parallel laufen dürfen. Die eigentliche *keepalived.conf* ist dabei auf beiden Servern identisch:


```

global_defs {
    notification_email {
        admin@example.com
    }
    notification_email_from admin@example.com
    smtp_server 192.168.10.54
    smtp_connect_timeout 30
    router_id IMAP-Cluster
}
vrrp_instance IMAP {
    state MASTER
    interface eth0
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass geheimespasswort
    }
    virtual_ipaddress {
        192.168.10.10
    }
}

```

Listing 10.50 Konfiguration in »etc/keepalived/keepalived.conf«

- ▶ Unter `global_defs` können Sie die Absender/Empfänger-Mailadresse und den SMTP-Server eintragen, damit Ihnen der *keepalived* Informationen über Statusänderungen senden kann.
- ▶ In der VRRP-Instanz mit dem selbst gewählten Namen `IMAP` werden das Netzwerkinterface (hier: `eth0`), eine auf beiden Systemen gleich zu wählende Cluster-ID (hier: `51`) und das auf beiden Systemen gleich einzutragende Passwort (hier: `geheimespasswort`) eingetragen.
- ▶ Zu guter Letzt wird bestimmt, dass dem jeweiligen Master zusätzlich zu seinen normal konfigurierten Systemadressen jeweils die Service-IP (hier: `192.168.10.10`) hinzugefügt wird.

Haben Sie mehrere VRRP-Server im gleichen Netzwerk aktiv, muss jedes Serverpärchen eine eigene `virtual_router_id` zugewiesen bekommen!



Ist Ihr Dovecot-Replikations-Cluster aktiv und sehen auch Ihre Tests mit *keepalived*/VRRP Erfolg versprechend aus, können Sie nun im DNS den IMAP-Hostnamen auf die hochverfügbar gehaltene Service-IP zeigen lassen, und schon sind Ihre Server hochverfügbar aufgestellt.

10.3 Anti-Spam/Anti-Virus mit Rspamd

Rspamd, maßgeblich programmiert von Vsevolod Stakhov, hat sich in den letzten 10 Jahren zum Shooting-Star der freien Anti-Spam-Systeme entwickelt und hat den (von uns all die Jahre äußerst geschätzten) Amavisd vielerorts abgelöst. Kein Wunder: Nicht nur die sehr moderne und vom ganzen Ansatz her sehr ausfallsichere und clusterbare Architektur, sondern auch die extrem vielen unterschiedlichen integrierten Ansätze zur Spam-Erkennung lassen des Postmasters Herz höher schlagen. Dazu kommt mit *Lua* eine mächtige integrierte Skriptsprache, die quasi alle Möglichkeiten eröffnet, den Rspamd-Cluster so individuell wie nur möglich zu konfigurieren und an die eigene Userlandschaft anzupassen.

10.3.1 Mails ablehnen oder in die Quarantäne filtern?

Vierelorten gilt es nach wie vor noch als gute Idee, dass E-Mails bei einem Spam- oder Virenbefund in einen Quarantäne-Ordner verschoben oder mit einer Spam-Markierung im Betreff zugestellt werden. Damit sollte zunächst vor allem das Problem gelöst werden, dass einmal angenommene E-Mails auch bei Spam- oder Virenbefund nicht einfach zum Absender zurückgesandt werden können, da die Absenderangaben dieser E-Mails typischerweise gefälscht sind und unschuldige Dritte sonst durch Millionen von E-Mail-Rückläufern außer Gefecht gesetzt werden würden. Wie bereits in Abschnitt 10.1.4 (im Punkt »Dynamische Empfängervalidierung«) erläutert wurde, sind *Backscatter*-Systeme eine echte Gefahr. Daher werden diese Systeme auch zu Recht in RBL-Sperrlisten aufgenommen. Doch wenn Sie die hier vorgestellte Kombination von Postfix und Rspamd nutzen, können Sie das Problem viel besser lösen: Im Handumdrehen installieren Sie damit ein System, das E-Mails schon während der Annahme, also in Echtzeit, nach Spam- und Virenmerkmalen durchsucht. Da die Mail sich noch in der Annahmephase befindet, kann sie nun im Falle eines Befundes noch gefahrlos abgelehnt werden.

Es handelt sich dabei um einen *REJECT* (die Mail wurde nie quittiert empfangen), was etwas völlig anderes ist als ein *Bounce* (die Mail wurde empfangen und kurz darauf zurückgesendet). Bei einem *REJECT* wird die Mail gegenüber dem ursprünglichen Client abgelehnt, sodass dieser sich überlegen muss, ob und wie er die Mail zurücksendet. *Backscatter*-Bounces würden also nicht bei Ihnen, sondern beim einliefernden System entstehen – und Spam-Botnetze, die versuchen, bei Ihnen besagte Mails mit gefälschten Absendern einzuliefern, würden natürlich gar kein Bounce erzeugen, sondern die fehlgeschlagene Einlieferung abrechnen. So schlagen Sie mehrere Fliegen mit einer Klappe:

- ▶ Sie sparen sich die Verwaltung und den Betrieb von Quarantäneverzeichnissen, denn erfahrungsgemäß schauen Nutzer in diese nicht hinein und/oder sind davon recht genervt.
- ▶ Sie müssen keine E-Mails mit Spam markiert zustellen, nur damit fast alle Nutzer diese E-Mails sowieso mithilfe eigener Filter ungelesen in Unterordner verschieben oder gar ungelesen löschen lassen.

Vor allem aber sorgen Sie dafür, dass im Falle eines *False Positives*, also einer fälschlicherweise gefilterten echten E-Mail, der Absender davon Kenntnis erhält, dass seine E-Mail den Empfänger nicht erreicht hat. Denn was ist die Realität? Versackt ein solcher False Positive in der Quarantäne, im Spam-Folder oder wird er gar ersatzlos gelöscht, erfahren weder der Absender noch der Empfänger davon, dass eine Mail nicht angekommen ist. Hilfreich ist das für die Beteiligten nicht, und es war in der Vergangenheit auch bereits Gegenstand gerichtlicher Auseinandersetzungen mit empfindlichen Schadenersatzzahlungen desjenigen, der seine Mails in der Spam-Quarantäne verloren hat.

Aber keine Angst: Auch mit dem hier vorgestellten Setup könnten Sie weiterhin Mails in die Quarantäne filtern oder im Betreff markieren lassen, denn auch bei einer Echtzeit-Filterung können Sie Rspamd anweisen, problematische Mails trotzdem zu akzeptieren und lediglich zu markieren. Sie können es – aber Sie müssen es nicht (mehr).

10.3.2 Installation von Rspamd, ClamAV und Redis

► Installation unter Debian/Ubuntu

Installieren Sie unter Debian/Ubuntu die benötigten Pakete:

```
mail:~ # apt install rspamd clamav-daemon clamav-freshclam redis
```

Listing 10.51 Die Installation unter Debian/Ubuntu

► Installation unter CentOS

Unter CentOS müssen Sie für die Installation von Redis auf das EPEL-Repository zurückgreifen und Rspamd aus dem Rspamd-eigenen Repository installieren:

```
root@linux:~# dnf install epel-release
root@linux:~# curl https://rspamd.com/rpm-stable/centos-8/rspamd.repo \
> /etc/dnf/repos.d/rspamd.repo
root@linux:~# rpm --import https://rspamd.com/rpm-stable/gpg.key
root@linux:~# dnf update
root@linux:~# dnf install rspamd clamav redis
```

Listing 10.52 CentOS: Installation von Rspamd, ClamAV und Redis

► Installation unter openSUSE

Bei openSUSE ist Rspamd in dem halboffiziellen Repository `server:mail` aktuell und gepflegt verfügbar. Sie müssen es gegebenenfalls erst hinzufügen, bevor Sie die Pakete mittels `zypper` installieren können:

```
root@linux:~# zypper ar https://download.opensuse.org/repositories/server:\
/mail/openSUSE_Leap_15.2/ server:mail
Adding repository 'server:mail' .....[done]
Repository 'server:mail' successfully added
```

```
URI      : https://download.opensuse.org/repositories/server:\
         /mail/openSUSE_Leap_15.2/
Enabled  : Yes
GPG Check : Yes
Autorefresh : No
Priority  : 99 (default priority)
```

Repository priorities are without effect. All enabled repositories share the same priority.

```
flash:~ # zypper in rspamd clamav redis
```

Listing 10.53 Installation von Rspamd unter openSUSE



Bevor Sie die neu installierten Dienste starten können, müssen Sie bei openSUSE zunächst eine Redis-Instanz aktivieren:

```
flash:~ # cp /etc/redis/default.conf.example /etc/redis/default.conf
```

Listing 10.54 Kleine Vorbereitungen für ClamAV und Redis unter openSUSE

Bei RedHat findet sich diese Datei als `/etc/redis.conf` wieder.

10.3.3 Update der Virensignaturen und Start der Dienste

Nachdem Sie ClamAV installiert haben, müssen als Allererstes aktuelle Virensignaturen heruntergeladen werden. Dies geschieht im Laufe der Zeit auch immer wieder über Cron-Jobs oder einen im Hintergrund laufenden Dämon. Aber zur Sicherheit sollten Sie einmal prüfen, ob `freshclam` meint, etwas tun zu müssen:

```
mail:~ # freshclam
ClamAV update process started at Wed Jun 14 15:43:44 2023
[...]
```

Listing 10.55 Der manuelle Start von »freshclam«

Anschließend müssen Sie alle soeben installierten Dienste einerseits »enablen«, sodass diese nach dem nächsten Reboot automatisch gestartet werden, und andererseits jetzt einmal (neu) starten, denn nicht jede Distribution macht das bei der Installation der Pakete automatisch:

► Für Debian, Ubuntu und CentOS wären die notwendigen Kommandos:

```
flash:~ # systemctl enable  rspamd clamav-daemon redis
flash:~ # systemctl restart rspamd clamav-daemon redis
```

Listing 10.56 Aktivierung der benötigten Dienste unter Debian/Ubuntu und CentOS

- ▶ Bei openSUSE hingegen erfolgt der Start von Redis etwas anders – Sie müssen hier die oben eingerichtete Instanz mit angeben:

```
flash:~ # systemctl enable  rspamd clamd freshclam redis@default
flash:~ # systemctl restart rspamd clamd freshclam redis@default
```

Listing 10.57 Aktivierung der benötigten Dienste unter openSUSE

10.3.4 Die Architektur von Rspamd

Der `rspamd`-Daemon ist ein eigenständig laufender Dämon (*Rspamd Worker*), der eine *HTTP Rest API* zur Verfügung stellt und so per TCP/IP angesprochen werden kann. Auf den eigentlichen Mailservern läuft dann ein kleiner *Rspamd Proxy*, ein schlanker Client ohne Filter-Logik, der über die standardisierte Mailserver-Schnittstelle *Milter* an Postfix oder Sendmail angebunden wird. Er erhält vom Mailserver die zu prüfende Mail samt Metadaten und gibt diese per HTTP-Call an die REST-API von Rspamd weiter. So können Sie einen zentralen Filter-Server sehr einfach von vielen verschiedenen Mailservern oder Mailrelays aus ansprechen und zentral und kraftvoll die notwendigen CPU-Ressourcen zur Spamfilterung vorhalten. Auf Ihrem Spamfilterserver öffnet Rspamd darum mehrere Dienste und Ports:

- ▶ **Rspamd Proxy** – TCP-Port 11332
nimmt Verbindungen vom Postfix per Milter-Protokoll entgegen.
- ▶ **Rspamd Worker** – TCP-Port 11333
ist der eigentliche Rspamd-Prozess und nimmt Verbindungen von seinen Proxy-Clients an. In komplexeren Setups könnte er sich auch auf einem anderen Server als dem eigentlichen Mailserver befinden.
- ▶ **Rspamd Controller** – TCP-Port 11334
Der Rspamd-Controller ist das Webinterface für den Admin und bietet eine Rest-API, um ihn in andere Prozesse einzubinden.

Nach der Installation bindet sich der Rspamd-Controller an `localhost:11334` und wäre über einen lokalen Webbrowser erreichbar. Um ihn auch von anderen Rechnern im Netz erreichbar zu machen, müssen Sie ein Passwort generieren und dessen (hier beispielhaft aufgeführten) Hash in der neu zu erstellenden Datei `local.d/worker-controller.inc` abspeichern und auch den Worker anweisen, auf »*« statt auf `localhost` zu horchen:

```
root@linux:~# rspamadm pw
Enter passphrase: <hidden input>
$1$cymbjp37q4w63iogc4erncz1tgm1ce9i5$kkfx9xc1wk9uuakqgx8kb
```

```
root@linux:~# vi /etc/rspamd/local.d/worker-controller.inc
password = "$1$cymbjp37q4w63iogc4erncz1tgm1ce9i5$kkfx9xc1wk9uuakqgx8kb";
bind_socket = "*:11334";
```

Listing 10.58 Rspamd-Admin-Passwort setzen und Webinterface nach außen öffnen

Loggen Sie sich dann direkt über <http://localhost:11334> in Ihr Webinterface ein (siehe Abbildung 10.1).

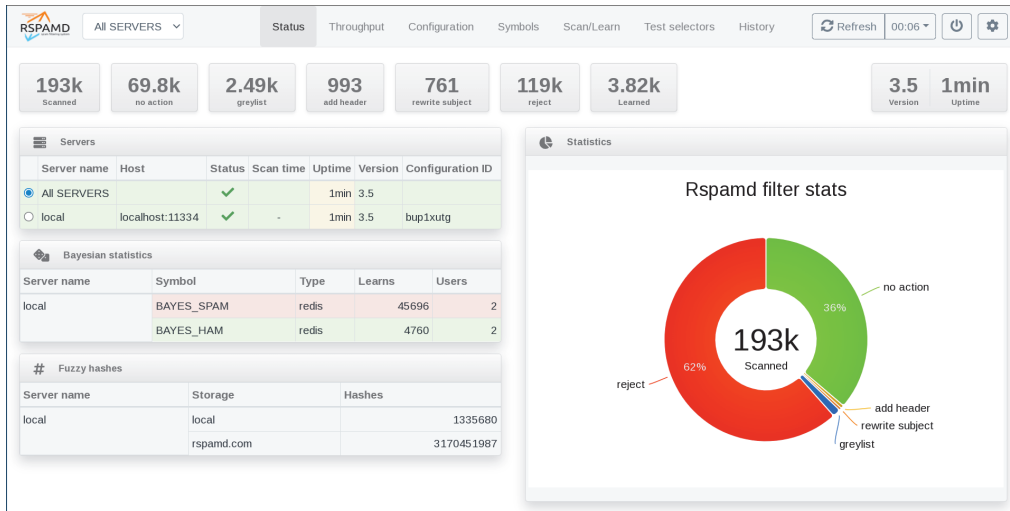


Abbildung 10.1 Das Webinterface von Rspamd nach dem Login



Sicher per SSH-Tunnel auf das Webinterface zugreifen

Der Controller unterstützt leider keine HTTPS-Verbindungen, sodass ein Zugriff aus dem öffentlichen Netzwerk problematisch sein kann. Sie können das Webinterface mit einem als HTTP-Proxy vorgeschalteten SSL-tauglichen Apache/Nginx absichern oder auf die Öffnung ins Internet doch kurzerhand verzichten. Für den nur gelegentlichen Zugriff auf das Webinterface können Sie SSH nutzen, um einen lokalen TCP-Port 11334 auf Ihrem Arbeitsrechner an den entfernten Port 11334 auf dem Rspamd-Worker weiterzuleiten. Bauen Sie dafür eine SSH-Verbindung zum Worker wie folgt auf:

```
ssh -L11334:localhost:11334 root@rspamdhost.example.com
```

Solange diese SSH-Verbindung steht, können Sie in Ihrem lokalen Webbrowser über die URL <http://localhost:11334> sicher über einen SSH-Tunnel verschlüsselt auf den entfernten Rspamd-Dienst zugreifen.

10.3.5 Einbindung von Rspamd an Ihren Postfix-Mailserver

Am besten binden Sie Rspamd über die MILTER-Schnittstelle an Postfix an. Dabei gibt Postfix die Mail ergänzt um einige Meta-Daten an den Rspamd-Proxy zur Prüfung und erhält von diesem anschließend Rückmeldung, ob die Mail angenommen werden darf oder nicht. Anders als bei den früher üblichen Amavis-Setups über `smtpd_proxy_filter` oder `content_filter`

verlässt die E-Mail dabei routingtechnisch nicht das Postfix-System, es wird lediglich nach einer Entscheidung gefragt. Der Postfix-Parameter `smtpd_milters` bindet Rspamd per Milter an alle Verbindungen des `smtpd` an, der Parameter `non_smtpd_milters` sorgt dafür, dass auch die auf Shell-Ebene über lokale Kommandos eingespeisten E-Mails gefiltert werden. Dazu können Sie die Parameter direkt in der `main.cf` global setzen:

```
root@linux:~# postconf -e "smtpd_milters=inet:12.7.0.0.1:11332"
root@linux:~# postconf -e "non_smtpd_milters=inet:127.0.0.1:11332"
```

Listing 10.59 Generische Anbindung über Milter-Aufrufe in der `main.cf`

Möchten Sie Rspamd nur selektiv an einen `smtpd`-Listener binden, können Sie das auch in der `master.cf` als Option für einen `smtpd` konfigurieren. Eine solche Konfiguration kann beispielsweise notwendig sein, wenn Ihre E-Mail mehrfach durch das Postfix-System geschleust wird, weil beispielsweise noch ein Crypto-Gateway beteiligt ist:

```
root@linux:~# vi master.cf
smtp      inet  n       -       n       -       -       smtpd
  -o smtpd_milters=inet:12.7.0.0.1:11332
```

Listing 10.60 Selektive Anbindung über einen Eintrag in der »`master.cf`«

Vergessen Sie nicht, Ihre Konfigurationsänderung anschließend per `postfix reload` zu aktivieren. Auch ohne weitergehende Konfiguration von Rspamd ist dieser nun bereits startklar und kann mit seinen Default-Einstellungen erfolgreich Spam filtern. Um zu sehen, ob alles richtig funktioniert, sollten Sie nicht nur eine normale Testmail senden, sondern dabei auch eine sogenannte GTUBE-Testmail verwenden, die ein Virenpattern simuliert und von Ihrem Rspamd korrekterweise erkannt und geblockt werden sollte. Auf <https://de.wikipedia.org/wiki/GTUBE> finden Sie eine Cut-and-paste-Vorlage für eine GTUBE-Testmail. Speichern Sie diese in eine Datei, und senden Sie sie an eine Testadresse:

```
root@linux:~# vi gtube-testmail.txt
Subject: Test spam mail (GTUBE)
Message-ID: <GTUBE1.1010101@example.net>
Date: Wed, 23 Jul 2003 23:30:00 +0200
From: Sender <sender@example.net>
To: Recipient <recipient@example.net>
Precedence: junk
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
```

```
This is the GTUBE, the
Generic
Test for
Unsolicited
```

Bulk
Email

If your spam filter supports it, the GTUBE provides a test by which you can verify that the filter is installed correctly and is detecting incoming spam. You can send yourself a test mail containing the following string of characters (in upper case and with no white spaces and line breaks):
XJS*C4JDBQADN1.NSBN3*2IDNEN*GTUBE-STANDARD-ANTI-UBE-TEST-EMAIL*C.34X

You should send this test mail from an account outside of your network.

```
root@linux:~# cat gtube-testmail.txt | sendmail testaccount@example.com
```

Listing 10.61 So versenden Sie eine GTUBE-Testmail.

Und so sollte sich die Ablehnung Ihrer Testmail im Logfile wiederfinden:

```
2021-02-13T08:59:40.504147+01:00 flash postfix/pickup[18747]: 7B08A3640215: \
uid=0 from=<root>
2021-02-13T08:59:40.505708+01:00 flash postfix/cleanup[18812]: 7B08A3640215: \
message-id=<GTUBE1.1010101@example.net>
2021-02-13T08:59:40.508477+01:00 flash postfix/cleanup[18812]: 7B08A3640215: \
milter-reject: END-OF-MESSAGE from localhost[127.0.0.1]: 5.7.1 Gtube pattern;\
from=<root@localhost> to=<p.heinlein@jpberlin.de>
2021-02-13T08:59:40.514805+01:00 flash postfix/cleanup[18812]: 7B08A3640215: \
to=<testaccount@example.com>, relay=none, delay=0.02, delays=0.02/0/0/0, dsn=5.7.1,\
status=bounced (Gtube pattern)
2021-02-13T08:59:40.515728+01:00 flash postfix/cleanup[18808]: 7DBC73640225: \
message-id=<20210213075940.7DBC73640225@localhost>
2021-02-13T08:59:40.521709+01:00 flash postfix/bounce[18811]: 7B08A3640215: \
sendernon-delivery notification: 7DBC73640225
```

Listing 10.62 Eine erfolgreiche Erkennung des GTUBE-Patterns.

10.3.6 Konfiguration des Rspamd

Nach der Installation von Rspamd finden Sie die Default-Konfigurationen in `/etc/rspamd` – über `/etc/rspamd/rspamd.conf` werden alle anderen Unterdateien per Include nachgeladen. Rspamd verwendet dabei eine Konfigurationssprache namens UCL, die *Universal Configuration Language*, die Ihnen auch durch den Webserver [nginx bekannt sein könnte. UCL ist eine Art lesbare Fassung eines JSON-Files und auf <https://rspamd.com/doc/configuration/ucl.html> ausführlich beschrieben. Es handelt sich dabei nicht um eine lineare Config-Datei, sondern sie kennt gewisse Strukturen: Parameter werden in eigenen Sektionen konfiguriert, die durch geschweifte Klammern begrenzt werden, wie in diesem Beispiel zu sehen ist:


```

dns {
    timeout = 1s;
    sockets = 16;
    retransmits = 5;
}

```

Listing 10.63 Beispiel für Sektionen in der Datei »options.inc«

Die wichtigsten Konfigurationsverzeichnisse von Rspamd sind:

- ▶ **/etc/rspamd**
ist das Hauptkonfigurationsverzeichnis: Es enthält die allgemeine Konfiguration des Rspamd-Dämons und vieler Module.
- ▶ **/etc/rspamd/modules.d**
definiert zusätzliche Module und Checks (*Plug-ins*), gegen die die E-Mails geprüft werden sollen.
- ▶ **/etc/rspamd/scores.d**
definiert die *Scores*, also welchen Zahlenwert ein einzelner Check zum Endergebnis beitragen soll.
- ▶ **/etc/rspamd/maps.d**
Normalerweise führt Rspamd viele Online-Zugriffe ins Internet aus: RBL-Abfragen und Abgleiche mit Bayes- oder Razor-Datenbanken. Dabei greift er auch auf spezielle Rspamd-Datenbanken zurück, die auf *maps.rspamd.de* bereitgestellt werden. Doch auch offline kann Rspamd eingesetzt werden, falls der Zugriff nicht erwünscht oder temporär nicht möglich ist. In diesem Fall wird hilfsweise auf bereitgehaltene Listen und Tabellen in diesem Verzeichnis zurückgegriffen, zum Beispiel für eine statische Liste bekannter Free-mail-Systeme oder bekannte DKIM-Whitelisting.

Wie es heutzutage zum guten Ton gehört, sollten Sie nicht die mitgelieferten Konfigurationsdateien editieren, da diese beim nächsten Paket-Update überschrieben werden könnten. Stattdessen bietet Ihnen Rspamd gleich zwei Verzeichnisse für eigene Konfigurationen an:

- ▶ **/etc/rspamd/local.d**
Der Unterordner *local.d* ist der Ort für eigene Konfigurationen, die die Default-Konfiguration *ergänzen* sollen. Die dort gefundenen Inhalte erweitern die eigentlich bereits vordefinierten Sektionen, die Parameter werden also hinzugefügt (*merge*). Üblicherweise ist es das, was Sie erreichen wollen.
- ▶ **/etc/rspamd/override.d**
In *override.d* hingegen können Sie eigene Konfigurationsschnipsel platzieren, die die jeweilige bereits vordefinierte Sektion vollständig *ersetzen* sollen. Dieser Ort ist also nicht geeignet, um einen einzelnen Wert zu konfigurieren, sondern Sie müssten hier jeweils die vollständige Sektion aufführen. Das benötigen Sie nur in Einzelfällen, beispielsweise wenn Sie eine komplette Sektion vollständig abschalten und »leer« setzen wollen.

Mit diesem Vorwissen können wir nun an den verschiedenen Stellen von Rspamd in die Detailkonfiguration einsteigen.

10.3.7 Konfiguration von Upstream-Quellen

An verschiedenen Stellen greift Rspamd auf andere Server und Dienste zurück, zum Beispiel:

- ▶ wenn der Rspamd-Proxy einen **Rspamd-Worker** auswählen soll,
- ▶ bei der Auswahl eines anzufragenden **Redis-Servers**,
- ▶ beim Zugriff auf einen **ClamAV-Dämon**, um eine Datei prüfen zu lassen,
- ▶ oder auch um DNS-Anfragen an einen **DNS-Resolver** weiterzuleiten.

Überall dort, wo Sie diese *Upstreams* konfigurieren, können Sie mehrere IP-Adressen, Hostnamen oder Unix-Sockets gleichzeitig verwenden und Rspamd mitteilen, nach welchem Verteilungsalgorithmus er diese Ziele nutzen soll:

- ▶ **master-slave**
spricht nur die erste Destination an, lediglich im Fehlerfall wird auf die weiteren angegebenen Destinationen zurückgegriffen.
- ▶ **round-robin**
spricht alle Destinationen entsprechend Ihrer Gewichtung zuverlässig der Reihe nach an.
- ▶ **random**
wählt nach dem Zufallsprinzip eine der Destinationen aus.
- ▶ **sequential**
spricht alle Destinationen der Reihe nach an, ohne auf Gewichtungen zu achten.
- ▶ **hash**
spricht Destinationen zufällig, aber abhängig vom jeweiligen *Lookup Key* an, d. h., gleiche Anfragen werden zum gleichen Ziel weitergeleitet.

Den gewünschten Algorithmus können Sie dabei Ihrer Liste voranstellen, wie in diesem Beispiel gezeigt:

```
# Ist alles okay, wird ausschließlich der erste Eintrag 192.168.10.11 verwendet:  
master-slave: 192.168.10.11,10.0.0.13,10.0.0.14
```

```
# Alle Hosts werden genutzt, allerdings wird 192.168.10.11 durch die Angabe  
# des angehängten ">:4"< nun viermal so stark gewichtet:  
round-robin: 192.168.10.11:4,10.0.0.13,10.0.0.14
```

Listing 10.64 Definition von Upstream-Ressourcen



Die Konfiguration von Upstreams wird etwas ausführlicher unter <https://rspamd.com/doc/configuration/upstream.html> erklärt.

10.3.8 Redis als schnelle Datenbank an der Seite von Rspamd

Redis ist ein einfaches Datenbanksystem, in dem sehr effektiv und extrem schnell Schlüssel-Wert-Kombinationen gespeichert werden können. In Abschnitt 10.3.2 wurde es bereits mitinstalliert. Rspamd nutzt Redis sehr intensiv als Cache, aber auch um langfristig zu lernen. Auch wenn es grundsätzlich möglich wäre, Rspamd ohne Redis zu betreiben, sollten Sie ihm unbedingt eine Redis-Datenbank an die Seite stellen.⁷

Das notwendige Redis-Paket haben wir in Abschnitt 10.3.2 bereits installiert, nun muss es noch in Rspamd eingebunden werden. Ein rein lokal betriebenes Redis könnten Sie gegebenenfalls auch passwortfrei ansprechen, doch spätestens im freien Internet müssen Sie Ihre Redis-Datenbank sowieso mit einem Passwort sichern.

Definieren Sie im ersten Schritt ein Redis-Passwort in `/etc/redis/default.conf`:

```
root@linux:~# /etc/redis/default.conf
requirepass some_secret_password
```

Listing 10.65 Definition eines Redis-Passworts

Erzeugen Sie dann für Rspamd die Datei `/etc/rspamd/local.d/redis.conf`, und verweisen Sie auf den lokalen Redis-Server:

```
root@linux:~# vi /etc/rspamd/local.d/redis.conf
servers = "127.0.0.1";
password = "some_secret_password";
```

Listing 10.66 Konfiguration des Redis-Zugriffs in Rspamd

10.3.9 Die Definition auszulösender Aktionen

Wir sind noch nicht so weit, unsere E-Mails nach Spam oder Viren filtern zu können. Trotzdem sollten wir uns schon jetzt einen Überblick verschaffen, was passieren könnte, wenn sich ein entsprechender Verdacht ergibt.

Folgende Aktionen sind (in aufsteigender Intensität) möglich:

► **greylist (»soft reject«)**

Die E-Mail wird mit einem temporären SMTP-Fehler, also einem 4xx-Fehlercode, zurückgewiesen. Dies entspricht einer Greylisting-Ablehnung. Das einliefernde Mailsystem wird die Mail darum in seiner Queue behalten und nach einigen Minuten eine erneute Zustellung versuchen.

⁷ Für einfache Szenarien reicht wie hier gezeigt ein auf dem Rspamd-Server installiertes lokales Redis; in komplexen großen Setups mit vielen Rspamd-Workern parallel sollten Sie auch Redis ebenfalls clustern und über mehrere Redis-Nodes bereitstellen – doch das sprengt leider den Rahmen dieses Kapitels.

► add_header

Rspamd wird die E-Mail zwar annehmen, aber den zusätzlichen von SpamAssassin-Systemen bekannten Mailheader `X-Spam: yes` setzen, den Sie gegebenenfalls später in nachgelagerten Systemen als Filterkriterium verwenden können. Rspamd selbst wird diese E-Mail in seiner eigenen Statistik bereits als Spam zählen.

► rewrite subject

Rspamd wird die E-Mail annehmen, aber eine entsprechende Spam-Markierung im Subject einfügen.

► reject

Die Mail wird von Rspamd mit einem fatalen 5xx-Fehlercode abgelehnt. Das einliefernde System wird die Mail daraufhin als unzustellbar zurück an den Absender senden.

► discard

Das Mailsystem wird angewiesen, die E-Mail zwar scheinbar mit einem »250 OK« erfolgreich anzunehmen, in Wirklichkeit aber ersatzlos zu löschen.

► quarantine

Die E-Mail soll in die HOLD-Queue des Mailsystems einsortiert werden, z. B. damit sie vom Administrator später individuell begutachtet werden kann.

Die Aktionen `discard` und `quarantine` sind nicht automatisch in Rspamd enthalten. Wenn Sie sie verwenden wollen, müssen Sie diese zuerst in der `/etc/rspamd/actions.conf` definieren:

```
actions {
# Used to send discard and quarantine commands to postfix
# Could be used as force action
discard = {
    flags = ["no_threshold"],
}
quarantine = {
    flags = ["no_threshold"],
}
```

Listing 10.67 »Discard« und »Quarantine« als neue Aktion im Rspamd

Welche Aktion ausgelöst wird, bestimmt der Spam-Score, den die E-Mail erreicht. Auch für den Spam-Score finden Sie Default-Werte in der Datei `/etc/rspamd/actions.conf`:

```
actions {
    reject = 15; # Reject when reaching this score
    add_header = 6; # Add header when reaching this score
    greylist = 4; # Apply greylisting when reaching this score (will emit
        # `soft reject action`)
    #subject = "***SPAM*** %s"
```

```
.include(try=true; priority=1; duplicate=merge) "$LOCAL_CONFDIR/local.d/\
actions.conf"
.include(try=true; priority=10) "$LOCAL_CONFDIR/override.d/actions.conf"
}
```

Listing 10.68 Die Schwellenwerte für Spam-Aktionen in Rspamd

Und auch, wenn wir sonst stets nach dem Grundsatz »Default-Werte sind gute Werte« handeln, möchten wir aufgrund unserer praktischen Erfahrung empfehlen, diese Werte doch anzupassen. Legen Sie dazu eine neue Datei `/etc/rspamd/local.d/actions.conf` mit eigenen Werten an. Da das Include auf diese Datei bereits aus dem actions-Block heraus erfolgte, dürfen Sie den action-Block hier nicht mehr erneut angeben:

```
root@linux:~# vi /etc/rspamd/local.d/actions.conf
reject = 15; # Reject when reaching this score
rewrite_subject = 13;
add_header = 10; # Add header when reaching this score
greylist = 8; # Apply greylisting when reaching this score (`soft reject action`)
subject = "[SPAM] %s"
```

Listing 10.69 Einrichtung eigener Schwellenwerte für Spam-Aktionen

10.3.10 Statistik und Auswertung im Webinterface

Wenn Sie wissen wollen, wie es Ihrem Spamfilter geht und was in der letzten Zeit passiert ist, bietet das Rspamd-Webinterface eine schöne Übersicht und eine tolle Darstellung. Dazu können Sie die letzten Scan-Ergebnisse als History in der Redis-Datenbank ablegen lassen.

Da beim Zugriff auf diese History jedoch sämtliche existierenden Objekte in den Webbrowser geladen werden, empfiehlt es sich, nicht mehr als 2.000 Datensätze zu speichern. So behalten Sie ein vertretbares Volumen und haben trotzdem eine in der Regel ausreichend lange History.

Normalerweise würde Rspamd auch den Betreff der jeweiligen E-Mails mit protokollieren. Mit `subject_privacy=true` können Sie das unterdrücken:



```
root@linux:~# vi /etc/rspamd/local.d/history_redis.conf
nrows = 2000;
compress = true;
subject_privacy = true;
```

Listing 10.70 Aktivierung der Rspamd-History im Webinterface

Benötigen Sie umfangreichere Auswertungen über längere Zeiträume? Dann wird die Rspamd-History für Sie zu limitiert sein. Werten Sie dann mit eigenen Skripten die Logfiles aus, oder speisen Sie Ihre Ergebnisse in Systeme wie ELK oder Clickhouse ein.

Im Webinterface einer laufenden Installation können Sie sich stets aktuelle Statistiken zu den Ergebnissen der letzten E-Mails ansehen. Aber auch in der Shell ist das kein Problem: Das Tool `rspamc` ist ein Client zur Webinterface-API und zeigt Ihnen exakt die gleichen Zahlen:

```
root@linux:~# rspamc stat
Results for command: stat (0.004 seconds)
Messages scanned: 516035
Messages with action reject: 467491, 90.59%
Messages with action soft reject: 0, 0.00%
Messages with action rewrite subject: 634, 0.12%
Messages with action add header: 888, 0.17%
Messages with action greylist: 2197, 0.42%
Messages with action no action: 44648, 8.65%
Messages treated as spam: 469013, 90.88%
Messages treated as ham: 46845, 9.07%
```

Listing 10.71 Statistiken: »`rspamc`« als CLI zur `Rspamd`-API

10.3.11 ClamAV in Rspamd einbinden

In Abschnitt 10.3.2 haben Sie ja bereits die Pakete des Virenkillers `ClamAV` installiert, aktualisiert und gestartet. Nun müssen Sie `Rspamd` nur noch anweisen, `ClamAV` für Virenprüfungen auch tatsächlich heranzuziehen:

```
root@linux:~# vi /etc/rspamd/local.d/antivirus.conf
clamav {
    servers = "127.0.0.1:3310";
    # Oder alternativ via UNIX-Socket:
    # servers = "/run/clamav/clamdctl";
}
```

Listing 10.72 Aktivierung von `ClamAV` in `Rspamd`

Würden wir die Sektion `clamav` um einen Eintrag `action=reject` erweitern, würde jede negative `ClamAV`-Rückmeldung sofort zur Ablehnung der fraglichen E-Mail führen, auch wenn `ClamAV` (nur?) ein Macro gefunden hat (`CLAM_VIRUS_MACRO`) oder die Mail ein verschlüsseltes und darum für `ClamAV` nicht scanbares Attachment beinhaltet (`CLAM_VIRUS_ENCRYPTED`). Scheitert (warum auch immer) der Virenkiller, verzichten `Rspamd` standardmäßig auf den Virencheck und lässt die E-Mail durch. Doch auch dies lässt sich individuell anpassen, wenn Sie bei dieser Gelegenheit auch gleich eine Sektion `CLAM_VIRUS_FAIL` definieren und bei Bedarf ein `soft reject` auslösen. Besser ist es darum, die verschiedenen `ClamAV`-Rückmeldungen über die sogenannten *forced actions* individuell mit Aktionen zu belegen:

```
root@linux:~# vi /etc/rspamd/local.d/force_actions.conf
rules {
```

```

VIRUS_SCANNER_FAIL_EXC {
    action = "soft reject";
    expression = "CLAM_VIRUS_FAIL";
    message = "Tempfail - internal scan engine error. (support-id: ${queueid})";
    # keinen Soft Reject wg. AV Fehlern, wenn eh schon rejected wird
    honor_action = ["reject"];
}
VIRUS_REJECT {
    action = "reject";
    expression = "CLAM_VIRUS";
    message = "REJECT - Virus found (support-id: ${queueid})";
}
VIRUS_REJECT_MACRO {
    action = "reject";
    expression = "CLAM_VIRUS_MACRO";
    message = "REJECT - attachment with macro found (support-id: ${queueid})";
}
# Keine Ablehnung bei verschlüsselten Archiven/Attachments
VIRUS_REJECT_ENCRYPTED {
    # action = "reject";
    expression = "CLAM_VIRUS_ENCRYPTED";
    message = "REJECT - encrypted attachment found (support-id: ${queueid})";
}
}

```

Listing 10.73 Abgestufte Reaktionen je nach ClamAV-Ergebnis

Ob Sie tatsächlich verschlüsselte E-Mails ablehnen wollen, weil Sie deren Inhalt nicht mit ClamAV filtern können, müssen Sie selbst entscheiden. In der *clamd.conf* (CentOS: *scan.conf*) können Sie aktivieren, dass ClamAV diese Funde meldet. Vergessen Sie nicht, anschließend die jeweiligen Dienste neu zu starten.

```

root@linux:~# vi /etc/clamd.conf
AlertEncrypted true
AlertEncryptedArchive true
AlertEncryptedDoc true
AlertOLE2Macros true

```

Listing 10.74 Richten Sie ein, was ClamAV melden soll.

10.3.12 Späteres Filtern über Mail-Header

Normalerweise kann Rspamd die gewünschten Aktionen wie Greylisting oder die finale Ablehnung einer E-Mail selbst auslösen. Das hat am Ende den großen Vorteil, dass Spam gar

nicht erst angenommen, sondern direkt im SMTP-Prozess abgelehnt wird. Die Einrichtung eines Spamverdachtsordners oder einer anderen Quarantäne ist dann unnötig. Wenn Sie Mails stattdessen lieber durch (unsichtbare) Mail-Header markieren wollen, um diese später irgendwohin sortieren zu können, geht das natürlich auch. Standardmäßig würde Rspamd bei der Aktion `add_header` jedoch nur den Eintrag `X-Spam: Yes` setzen. Über `use=` können Sie einzelne Mailheader ergänzen lassen bzw. über `extended_spam_headers=true` ein Set der üblichen Spam-Header ergänzen lassen:

```
# Routines to use- this is the only required setting (may be omitted if using
# extended_spam_headers)
use = ["x-spamd-bar", "x-spamd-level", "authentication-results"];

# Add "extended Rspamd headers" (default false) (enables x-spamd-result,
# x-rspamd-server & x-rspamd-queue-id routines)
extended_spam_headers = true;

# Set false to always add headers for local IPs (default true)
skip_local = true;

# Set false to always add headers for authenticated users (default true)
skip_authenticated = true;
```

Listing 10.75 E-Mails Spam-Header hinzufügen in »`milter_headers.conf`«

Und damit Rspamd weiß, welche Mails von vertrauenswürdigen lokalen Clients stammen, müssen Sie die Definition der lokalen Netzwerke aktuell halten:

```
# Local networks
local_addrs = [192.168.0.0/16, 10.0.0.0/8, 172.16.0.0/12, fd00::/8, 169.254.0.0/16,\
  fe80::/10];
```

Listing 10.76 Definition von lokalen, vertrauenswürdigen Netzwerken in »`options.inc`«



Milter-Header sind unter https://rspamd.com/doc/modules/milter_headers.html ausführlicher beschrieben.

10.3.13 RBLs in Rspamd

Wie DNS-Blacklists (RBL) funktionieren, wurde bereits in Abschnitt 10.1.4 erklärt. Rspamd bringt natürlich auch ein eigenes RBL-Plug-in mit, das bereits vorkonfiguriert die wichtigsten RBL-Listen abfragt:

- ▶ `zen.spamhaus.org` und `dbl.spamhaus.org`
- ▶ `rspamd.com`

- ▶ URIBL.com
- ▶ senderscore.com
- ▶ spameatingmonkey.net
- ▶ dnswl.org
- ▶ virusfree.cz
- ▶ ix.dnsbl.manitu.net
- ▶ blocklist.de
- ▶ msbl.org
- ▶ surbl.org

Einige dieser Listen unterliegen, wie in Abschnitt 10.1.4 erläutert, einer *Fair-Use-Policy* und sind nur für den nichtkommerziellen Einsatz bei kleinem Abfragevolumen frei zu nutzen.

Gerade Spamhaus geht seit einigen Jahren sehr systematisch gegen hohe Nutzungsvolumen durch nicht lizenzierte Anwender vor. Haben Sie eine Spamhaus-Lizenz erworben, so besitzen Sie eine eigene gehashte RBL-Subdomain, gegen die Ihre Abfrage läuft, beispielsweise `ohzxxxxxxxxxxxxq2ky.zen.dq.spamhaus.net`. In diesem Fall müssen Sie nun dafür sorgen, dass Ihr Rspamd-System diese Domain und nicht mehr `zen.spamhaus.org` verwendet:

```

rbls {
    spamhaus {
        rbl = "ohzxxxxxxxxxxxxq2ky.zen.dq.spamhaus.net";
    }
    "DBL" {
        rbl = "ohzxxxxxxxxxxxxq2ky.dbl.dq.spamhaus.net";
    }
}

```

Listing 10.77 Eigene RBL-Domains setzen (Lizenzen erforderlich) in »rbl.conf«

In einigen Fällen kann es auch notwendig sein, RBL-Domains kurzfristig oder vorübergehend abzuschalten, z. B. bei einer akuten Störung:

```

rbls {
    spamhaus {
        disabled = true;
    }
    "DBL" {
        disabled = true;
    }
}

```

Listing 10.78 Deaktivierung einzelner RBLs in »rbl.conf«



Achtung: Die Datei `/etc/rspamd/local.d/rbl.conf` wird am Ende der Datei `/etc/rspamd/modules.d/rbl.conf` bereits *innerhalb* der Sektion `rbl` aufgerufen. Dadurch entsteht der häufige Fehler, dass in `/etc/rspamd/local.d/rbl.conf` nicht noch mal die umschließende `rbl`-Sektion gelistet werden darf.

```
root@linux:~# vi /etc/rspamd/modules.d/rbl.conf
rbl {
    rbls {
        [...]
    }
    [...]
    .include(try=true,priority=5) "$DBDIR/dynamic/rbl.conf"
    .include(try=true,priority=1,duplicate=merge) "$LOCAL_CONFDIR/local.d/rbl.conf"
    .include(try=true,priority=10) "$LOCAL_CONFDIR/override.d/rbl.conf"
}
```

Listing 10.79 Beachten Sie, in welchem Kontext »`local.d/rbl.conf`« eingebunden wurde.

10.3.14 Bayes in Rspamd

Mithilfe sogenannter *Bayes*-Filter erkennen Anti-Spam-Systeme Wortzusammenhänge und Phrasen. Dadurch können Spam-Mails auch dann als bekannte Massenmail erkannt werden, wenn der Inhalt vom Spammer in den E-Mails jeweils leicht modifiziert wird. Normalerweise müssen Bayes-Filter mit »guten« und »bösen« E-Mails (*Ham* bzw. *Spam*) trainiert werden, was in der Praxis leider oft zu sehr schlechten Ergebnissen führt, da viele Administratoren zwar fleißig »böse« Spam-Mails trainieren, aber schnell vergessen, mit ausreichend »guten« Mails die Gegenbeispiele zu setzen.

Besser ist es darum, Rspamd das Training seines Bayes-Filters selbst zu überlassen: Dabei werden E-Mails, die aus anderen Gründen als Spam klassifiziert werden, als Spam-Mails registriert und gleichzeitig auch E-Mails, die ganz klar kein Spam sind, als Ham gelernt. Aktivieren Sie darum den Autolearn-Modus, und freuen Sie sich, dass das System damit sehr pflegefrei *out-of-the-box* funktionieren wird:

```
root@linux:~# vi /etc/rspamd/local.d/statistics.conf
classifier "bayes" {
    autolearn {
        spam_threshold = 17.0; # When to learn spam (score >= threshold)
        ham_threshold = -0.5; # When to learn ham (score <= threshold)
        check_balance = true; # Check spam and ham balance
        min_balance = 0.9; # Keep diff for spam/ham learns for at least this value
    }
}
```

Listing 10.80 Einrichtung des Autolearn-Modus für das Bayes-Modul »`statistics`«

Unter der URL <https://rspamd.com/doc/configuration/statistic.html> ist das statistics-Modul ausführlich beschrieben.



10.3.15 Eigene White- und Blacklists führen

Um White- oder Blacklists zu pflegen, gibt es verschiedene Wege in Rspamd. Wir möchten Ihnen hiermit nahelegen, dies flexibel über das Modul `multimap` zu lösen. Sie können damit Textdateien mit sehr einfachen Einträgen pflegen, behalten für fortgeschrittene Setups aber auch die Möglichkeit, später über sogenannte *Composites* verschiedene Filterbedingungen zu verknüpfen. *Multimaps* sind lokale Datenquellen im Rspamd; das können einfache Dateien mit Texteinträgen oder Regexp-Pattern sein; in fortgeschrittenen Setups können auch HTTP-Requests oder Redis-Datenbanken genutzt werden.

Schon mit diesem Beispiel können Sie unkompliziert Domains *soft* white- oder blacklisten, indem Sie den E-Mails einen Spam-Score von +8.0 bzw. -8.0 zuweisen:

```
# Achtung From ist ENV-From
SENDER_DOMAIN_WHITELIST {
    type = "from";
    filter = "email:domain";
    map = "/etc/rspamd/local.d/maps.d/sender_domain_whitelist.map";
    score = -8.0;
}
SENDER_DOMAIN_BLACKLIST {
    type = "from";
    filter = "email:domain";
    map = "/etc/rspamd/local.d/maps.d/sender_domain_blacklist.map";
    score = 8.0;
}
```

Listing 10.81 Soften White- und Blacklists per Multimap in »multimap.conf«

Anstatt einen Scores von +8 bzw. -8 Punkten zu definieren, können Sie auch direkt eine finale harte Aktion auslösen lassen:

```
# Achtung From ist ENV-From
SENDER_DOMAIN_WHITELIST {
    type = "from";
    filter = "email:domain";
    map = "/etc/rspamd/local.d/maps.d/sender_domain_whitelist.map";
    action = accept;
    prefilter = true;
}
SENDER_DOMAIN_BLACKLIST {
    type = "from";
```

```
filter = "email:domain";
map = "/etc/rspamd/local.d/maps.d/sender_domain_blacklist.map";
action = reject;
prefilter = true;
}
```

Listing 10.82 Harten White- und Blacklists per Multimap in »multimap.conf«

Durch die Angabe von `filter = email:domain;` wird Rspamd angewiesen, nicht die komplette E-Mail-Adresse, sondern nur den Domainanteil als Suchkriterium zu verwenden. In den hier beispielhaft konfigurierten Dateien `sender_domain_whitelist.map` bzw. `sender_domain_blacklist.map` können Sie also nun zeilenweise Domains aufführen und damit white- bzw. blacklisten:

```
root@linux:~# vi /etc/rspamd/local.d/maps.d/sender_domain_whitelist.map
gutes.example.com
mydomain.example.org
```

```
root@linux:~# vi /etc/rspamd/local.d/maps.d/sender_domain_blacklist.map
spamquelle.example.com
phishing.example.org
```

Listing 10.83 Beispieleinträge in White- und Blacklists

Möchten Sie lieber einzelne Mailadressen aufführen, so lassen Sie die jeweilige `filter`-Einstellung weg. Ganz analog können Sie übrigens auch unliebsame Dateianhänge blockieren lassen:

```
root@linux:~# vi /etc/rspamd/local.d/multimap.conf
BANNED_EXTENSIONS {
    type = "filename";
    filter = "extension";
    map = "/etc/rspamd/local.d/maps.d/banned_extensions.map";
    symbol = "BANNED_EXTENSIONS";
    action=reject;
    prefilter = true;
    message = "A restricted file type was found";
}
```

```
root@linux:~# vi /etc/rspamd/local.d/maps.d/banned_extensions.map
exe
com
scr
cmd
```

Listing 10.84 Blacklists für Dateianhänge

Auch RegExp-Pattern können in den Map-Dateien verwendet werden, solange die Option `regexp = true`; gesetzt wird. Sinnvoll kann das sein, wenn Sie eigene Subdomains matchen wollen oder Pattern benötigen, um bestimmte Localparts von Mailadressen zu erfassen. Verwenden Sie dabei übliche Regexp-Pattern mit Schrägstrichen als Pattern-Begrenzer. Dieses Beispiel definiert einen weiteren, zusätzlichen Lookup, der nicht mehr die Domain der Mailadresse (`filter=email:domain`), sondern direkt die Mailadresse des Absenders verwendet:

```
root@linux:~# vi /etc/rspamd/local.d/multimap.conf
SENDER_FROM_BLACKLIST {
    type = "from";
    map = "file:///etc/rspamd/local.d/maps.d/sender_from_blacklist.map";
    regexp = true;
    score = 8.0;
}
```

```
root@linux:~# vi /etc/rspamd/local.d/maps.d/sender_from_blacklist.map
/username@.*example.com/
/paypal.*.*/
```

Listing 10.85 Blacklists mit Regexp-Unterstützung

10.3.16 Einrichtung von DKIM zur Mailsignierung

Mittels *Domain Key Identified Mail* (DKIM) signieren Mailserver E-Mails, sodass andere Systeme über den im DNS veröffentlichten öffentlichen Schlüssel überprüfen können, ob eine E-Mail tatsächlich von dem fraglichen Mailserver stammt oder ob es sich dabei um eine Absenderfälschung handelt. Große Provider strafen in ihren Anti-Spam-Systemen Mails ohne DKIM-Signatur ab, sodass es sehr empfehlenswert ist, DKIM einzurichten. Das ist in Rspamd auch erfreulich einfach: Sie müssen nur einen DKIM-Schlüssel erzeugen lassen und den Public Key als DNS-Record veröffentlichen. Rspamd wird dann automatisch E-Mails signieren, sobald er für die Absender-Domain der E-Mail einen öffentlichen DKIM-Schlüssel findet, zu dem er den zugehörigen privaten Schlüssel besitzt.

Erzeugen Sie dafür als Erstes einen neuen DKIM-Key, mit dem Rspamd alle Nachrichten signieren kann:

```
root@linux:~# mkdir /var/lib/rspamd/dkim
root@linux:~# rspamadm dkim_keygen -s DKIM001 > /var/lib/rspamd/dkim/DKIM001.key
```

Listing 10.86 Erzeugen Sie einen eigenen DKIM-Schlüssel

Anschließend müssen Sie den Public Key Ihres neuen DKIM-Key im DNS veröffentlichen, damit er von anderen Mailservern abgerufen werden kann. Dafür ist in der soeben neu erzeugten Datei bereits ein Eintrag vorbereitet, den Sie per Cut-and-paste in Ihre DNS-Zonen übernehmen können, sodass er später unter `DKIM001._domainkey.example.com` abrufbar ist:

```
root@linux:~# less /var/lib/rspamd/dkim/DKIM001.key
-----BEGIN PRIVATE KEY-----
[...]
-----END PRIVATE KEY-----
DKIM001._domainkey IN TXT ( "v=DKIM1; k=rsa; "
    "p=MIGfMAOGCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDXzgnNAGg73NGB143rExx2nAjMqgL1DqBrhb
+WrUMUgJrIGz0teKq4JKc5bB517nTav9kWthshEUrywuBMOFwaocEK4F8MwexATePxArGBp9LhtSJiti4dNBQ
91IUu/gk+TDOKniNb00NKxCWnIMJkuilgeA26mTwEG+LC09ggEQIDAQAB" ) ;
```

Listing 10.87 Der erzeugte DKIM-Schlüssel zeigt auch den notwendigen DNS-Record.

Weisen Sie Rspamd nun an, alle ausgehenden E-Mails zu signieren. Sie können dabei dieses Konfigurationsbeispiel verwenden – achten Sie aber auf jeden Fall darauf, dass Sie stets den richtigen *Selektor* verwenden, hier DKIM001, passend zum Dateinamen und zum DNS-Hostnamen, unter dem Sie den Schlüssel veröffentlicht haben:

```
# Default selector to use
selector = "DKIM001";

# Default path to key, can include '$domain' and '$selector' variables
path = "/var/lib/rspamd/dkim/$selector.key";

# If false, messages from authenticated users are not selected for signing
sign_authenticated = true;

# If false, messages from local networks are not selected for signing
sign_local = true;

# If false, messages with empty envelope from are not signed
allow_envfrom_empty = true;

# If true, envelope/header domain mismatch is ignored
allow_hdrfrom_mismatch = true;

# If true, username does not need to contain matching domain
allow_username_mismatch = true;

# Whether to fallback to global config
try_fallback = true;

# Domain to use for DKIM signing: can be "header" (MIME From),
"envelope" (SMTP From) or "auth" (SMTP username)
use_domain = "header";
```

```
# If `true` get pubkey from DNS record and check if it matches private key
check_pubkey = true;

# Set to `false` if you want to skip signing if public and private keys mismatches
allow_pubkey_mismatch = false;
```

Listing 10.88 DKIM-Signierung in Rspamd einrichten in »dkim_signing.conf«

Dank `check_pubkey = true`; prüft Rspamd, ob der Public Key im DNS einer Domain verfügbar ist, und signiert nur, wenn alles richtig eingerichtet worden ist. In den Headerzeilen der fraglichen E-Mails sollten Sie nun einen längeren DKIM-Header vorfinden.

Den hier generierten DKIM-Key können Sie auch für mehrere Domains gleichzeitig verwenden – Sie müssen ihn nur unter dem jeweiligen Selektor im DNS der Domain veröffentlichen. Aber natürlich können Sie auch mehrere DKIM-Keys für eine Domain oder verschiedene DKIM-Keys für verschiedene Domains konfigurieren. Ergänzen Sie die Konfiguration dazu wie folgt:

```
root@linux:~# vi /etc/rspamd/local.d/dkim_signing.conf
domain {
  example.com {
    selectors [
      {
        path = "/var/lib/rspamd/dkim/example.com.dkim.key";
        selector = "dkim";
      },
      {
        path = "/var/lib/rspamd/dkim/ncxs.de.dkim-test.key";
        selector = "dkim-test";
      }
    ]
  }
  example.net {
    selectors [
      {
        path = "/var/lib/rspamd/dkim/example.net.main.key";
        selector = "main";
      }
    ]
  }
}
```

Listing 10.89 DKIM-Signierung in Rspamd einrichten

10.3.17 Ausblick: Einbindung weiterer Prüfungsmethoden

Rspamd ist ein mächtiges Framework, um alle möglichen verschiedenen Prüfungsmechanismen zur Viren- oder Spamprüfung einzubinden. Vieles ist in Rspamd bereits eingebaut, doch über die sogenannten *external services* können Sie Möglichkeiten noch einmal deutlich erweitern. Es sprengt leider den Rahmen dieses Buches, hier auf alles einzugehen, doch unter https://rspamd.com/doc/modules/external_services.html sind die weiteren Anbindungen, unter anderem von

- ▶ DCC,
- ▶ OleTools,
- ▶ Razor und
- ▶ SpamAssassin

übersichtlich erklärt. Durch die Emotet-Virenwellen der letzten Jahre sind dabei die sogenannten *OleTools* beliebt und bekannt geworden. Wir stellen sie in einer fünfteiligen Serie in unserem Blog <http://www.heinlein-support.de/blog/news/emotet-mit-rspamd-und-oletools-bekaempfen/> ausführlich vor und möchten Sie Ihnen wärmstens ans Herz legen.

10.4 Monitoring und Logfile-Auswertung

Die Überwachung eines Mailservers ist nicht ganz trivial. Es reicht nicht, zu prüfen, ob Port 25 geöffnet oder der IMAP-Server erreichbar ist. Bei der Verarbeitung einer E-Mail werden viele versteckte Arbeitsschritte durchlaufen, von denen jeder einzelne scheitern kann. Vergessen Sie auch nicht, dass Ihr Mailserver vielleicht wunderbar funktioniert, wegen Überlastung und Mailstau jedoch derart verzögert seine Mail-Queue abarbeitet, dass Sie einen Mailstau als Fehler in Ihrem Monitoring feststellen wollen, bevor sich die Nutzer beschweren. Zu einer Überwachung eines Mailservers gehören:

- ▶ die grundsätzliche Systemüberwachung wie bei jedem Server (RAM, Plattenplatz, Temperatur, I/O-Leistung, Connectivity etc.)
- ▶ die Prüfung der Verfügbarkeit aller gestarteten Dienste (Postfix, Dovecot, Rspamd, Clamd, Postgrey, Redis etc.)
- ▶ die Anzahl der Mails in der Postfix-Queue, also die Anzahl der Dateien in */var/spool/postfix/incoming* sowie in *active* und *deferred* (Stau?). Profis ermitteln das Alter der ältesten Datei in *incoming* und ziehen so Rückschlüsse auf die Dauer eines etwaigen Rückstaus.
- ▶ die Anzahl der laufenden *smtp*- bzw. *smtpd*-Prozesse (Überlastung?). Postfix startet per Default maximal 100 davon.

Dazu sei Ihnen allgemein das Kapitel 29, »Monitoring – wissen, was läuft«, ans Herz gelegt, denn mit dem dort vorgestellten Checkmk können Sie all das bequem überwachen.

Kapitel 11

Datenbank

In diesem Kapitel finden Sie Basisstrategien für die Aufgaben, denen Sie bei Datenbanken regelmäßig gegenüberstehen.

An Datenbanksystemen mangelt es in der Linux-Welt nicht. Die Entscheidung, in diesem Kapitel ausschließlich *MariaDB* zu behandeln, fiel aus zwei Gründen. Zum einen ist MariaDB mittlerweile das Linux-Datenbanksystem mit der weitesten Verbreitung in neuen Distributionen. Zum anderen ist es als Fork von MySQL spätestens seit der Version 5.0 endgültig den Kinderschuhen entwachsen und enthält Funktionen, die früher nur teuren kommerziellen Systemen vorbehalten waren.

11

11.1 MariaDB in der Praxis

Alle populären Linux-Distributionen enthalten MariaDB-Pakete, bei vielen wurde MySQL durch MariaDB ersetzt. In CentOS findet sich die Version 10.5.16, in Debian die Version 10.5.19, in Ubuntu und in openSUSE Leap die Version 10.6.12.

Bitte beachten Sie, dass wir – wenn nicht explizit auf eine andere Konfigurationsdatei verwiesen wird – alle Konfigurationen in der Datei *adminbuch.cnf* in dem Verzeichnis */etc/mysql/mariadb.conf.d* für Debian und Ubuntu bzw. im Verzeichnis */etc/my.cnf.d* unter CentOS und openSUSE Leap vornehmen.



11.1.1 Installation und grundlegende Einrichtung

Installieren Sie MariaDB mit dem folgenden Kommando Ihrer Distribution:

```
### Debian und Ubuntu  
apt install mariadb-server
```

```
### CentOS  
dnf install mariadb-server
```

```
### openSUSE  
zypper install mariadb
```

Listing 11.1 Installation von MariaDB

Abhängig von der verwendeten Linux-Distribution werden Sie möglicherweise bereits während der Installation nach einem Passwort für den MariaDB-Benutzer `root` gefragt. Er ist nicht identisch mit dem Superuser, also dem `root`-Account Ihres Linux-Systems, sondern repräsentiert lediglich den Administrator des MariaDB-Servers. In allen von uns beschriebenen Distributionen starten Sie den Datenbankserver mit `systemctl start mariadb`. Allerdings werden bei allen hier behandelten Distributionen die My-SQL-Clients, wie beispielsweise `mysql` und `mysqladmin`, verwendet. Sollte die Installationsroutine Sie nicht nach einem Passwort fragen, vergeben Sie es unmittelbar nach der Installation und dem Start des Servers mit dem folgenden Kommando:

```
/usr/bin/mysqladmin -u root password 'NeuesPasswort'
```

Listing 11.2 Passwort für den »root«-Account des MariaDB-Servers vergeben

Zusätzlich ist es sinnvoll, ein weiteres Administrationskonto einzurichten. Dazu loggen Sie sich zunächst als `root` auf dem MariaDB-Server ein:

```
mysql -u root -p
```

Listing 11.3 Login auf dem MariaDB-Server

Nach der erfolgreichen Eingabe des Passworts ändert sich Ihr Kommandozeilenprompt in MariaDB `[(none)]>`. Jetzt können Sie das neue Benutzerkonto anlegen:

```
GRANT ALL ON *.* TO 'ich'@'localhost' IDENTIFIED BY 'MeinKennwort' WITH GRANT OPTION;
```

Listing 11.4 Ein neues Benutzerkonto mit weitreichenden Rechten anlegen

Achten Sie besonders auf das abschließende Semikolon; es ist unbedingt erforderlich, wird aber häufig vergessen. Mit `exit` verlassen Sie den MariaDB-Kommandoprompt wieder.



Per Default bindet sich MariaDB unter Debian, openSUSE und Ubuntu unmittelbar nach der Installation nur an die IP-Adresse `127.0.0.1` (`localhost`). Wenn Sie diese Einstellung ändern möchten, erstellen Sie bitte unter Debian, openSUSE und Ubuntu eine Datei namens `/etc/mysql/mariadb.conf.d/adminbuch.cnf`, in die Sie unterhalb des Eintrags `[mysqld]` einen Eintrag `bind-address = 0.0.0.0` aufnehmen.



Mit der Einstellung `bind-address = 0.0.0.0` bindet sich MariaDB an alle auf dem Server konfigurierten IP-Adressen. Prüfen Sie vor einem solchen Schritt genau, ob diese Konfiguration notwendig und – gerade unter Sicherheitsgesichtspunkten – sinnvoll ist. Nach einem Neustart des Datenbankservers sollte Ihnen `lsof` Informationen analog zur folgenden Ausgabe liefern:

```
# lsof -i tcp:3306
COMMAND  PID  USER  FD  TYPE DEVICE SIZE/OFF  NODE NAME
mysqld  11735  mysql  14u  IPv4  28487      0t0  TCP *:mysql (LISTEN)
```

Listing 11.5 Feststellen, auf welche IP-Adressen der Datenbankserver hört

Wie Sie sehen können, heißt das Kommando trotz Abspaltung vom MySQL-Projekt immer noch `mysqld` und nicht wie erwartet `mariadb`.

11.1.2 Replikation

Durch Replikationsmechanismen können Sie Datenbanken, die sich auf unterschiedlichen MariaDB-Servern befinden, auf dem gleichen Stand halten.

Das bietet eine Reihe von Vorteilen. So können Sie beispielsweise die Last gezielt auf mehrere Server verteilen:

- ▶ Eine Datenbank erhält nur Aktualisierungen, eine andere nur Abfragen.
- ▶ Eine Datenbank erhält nur Aktualisierungen, mehrere andere erhalten Abfragen.
- ▶ Mehrere Datenbanken erhalten sowohl Aktualisierungen als auch Abfragen.

Mittels Replikation ist es auch möglich, eine Datenbank auf den Notebooks von Außendienstmitarbeitern automatisch mit einer Datenbank in der Firmenzentrale abzugleichen, damit die Mitarbeiter stets über den neuesten Datenbestand verfügen.

Wir werden folgende Replikationsmethoden betrachten:

▶ Master-Slave-Replikation

Ein Master-Server kann einen oder mehrere Slaves haben. Der Master speichert Updates auch in *Binlogs* (binären Log-Dateien). Wenn ein Slave eine Verbindung zum Master herstellt, teilt er ihm mit, bis zu welcher Position er die Binlogs beim letzten Update gelesen hat. Der Master repliziert daraufhin alle Änderungen, die seit diesem Zeitpunkt aufgelaufen sind, auf den Slave. Danach wartet der Slave, bis er vom Master über weitere Änderungen informiert wird. Während dieser Zeit kann er Anfragen beantworten, Aktualisierungen werden aber immer nur auf dem Master vorgenommen.

▶ Master-Master-Replikation (Ring-Replikation)

Hier dürfen Schreibvorgänge auf allen beteiligten Servern durchgeführt werden, aber das birgt auch die Gefahr, dass `auto_increment`-Felder auf allen Mastern gleichzeitig inkrementiert werden und so doppelte Werte auftreten. Daher müssen für diesen Fall Vorkehrungen getroffen werden.

Die erwähnten binären Logfiles zeichnen von dem Moment an, an dem das Logging gestartet wird, alle Änderungen an den Datenbanken des Master-Servers auf. Wenn Sie einen Replikationsverbund aufbauen, ist es wichtig, dass zum Start der Replikation alle Server den Stand der Master-Datenbanken zu dem Zeitpunkt haben, an dem die binären Logfiles aktiviert wurden. Weichen die Stände voneinander ab, wird es sehr wahrscheinlich zu Fehlern bei der Replikation kommen. Achten Sie auch auf mögliche Versionsunterschiede bei den MariaDB-Servern, die Sie einsetzen. Neuere Slaves können in der Regel mit älteren Mastern zusammenarbeiten, aber nicht umgekehrt.



Konfiguration eines Master-Servers

Legen Sie zunächst einen Benutzer für die Replikation an. Der Benutzer für das folgende Beispiel heißt `repl` und bekommt das Passwort `slavepass`, und der User darf sich aus dem angegebenen Netzbereich anmelden:

```
MariaDB [(none)]> GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.*
  -> TO repl@'192.168.1.%' IDENTIFIED BY 'slavepass';
```

oder

```
MariaDB [(none)]> GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.*
  -> TO 'repl'@'%.mydomain.com' IDENTIFIED BY 'slavepass';
```

Listing 11.6 Einen Benutzer für die Replikation anlegen

Vergewissern Sie sich nun, dass der Abschnitt `[mysqld]` der Datei `adminbuch.cnf` (siehe auch den Hinweis zu Beginn des Abschnitts 11.1) auf dem Master-Server die Option `log-bin` enthält. Der Abschnitt sollte außerdem eine Option `server-id=master_id` enthalten, wobei `master_id` ein positiver Integer zwischen 1 und 4294967295 ($2^{32} - 1$) sein muss. Da der Default-Wert 1 ist, kann er auch auf einem schlecht konfigurierten Server noch »versehentlich« gesetzt sein, deshalb sollten Sie einen anderen Wert wählen. Damit Sie die größtmögliche Dauerhaftigkeit und Konsistenz in einer Replikationskonfiguration für InnoDB¹ mit Transaktionen zu erzielen, sollten Sie in jedem Fall `innodb_flush_log_at_trx_commit=1` und `sync_binlog=1` in der Datei `adminbuch.cnf` auf dem Master angeben. So sieht ein Beispiel für den fertigen Konfigurationsabschnitt aus:

```
[mysqld]
log_bin                = /var/log/mysql/mariadb-bin.log
max_binlog_size        = 100M
server_id = 100
innodb_flush_log_at_trx_commit=1
sync_binlog=1
```

Listing 11.7 Die Änderungen in der Datei »adminbuch.cnf«



Nachdem Sie diese Änderungen vorgenommen haben, müssen Sie den MariaDB-Daemon neu starten. Falls Sie CentOS benutzen, legen Sie vorher noch mit den folgenden Kommandos das Verzeichnis `/var/log/mysql` für den User `mysql` an:

```
mkdir /var/log/mysql
chown mysql:mysql /var/log/mysql
```

Listing 11.8 Das Log-Verzeichnis anlegen und dem User »mysql« zuordnen

¹ InnoDB ist eine freie Storage Engine für MySQL/MariaDB, die sich insbesondere durch Transaktionssicherheit auszeichnet.

Nun muss der Slave erstmalig mit einer Momentaufnahme der Daten des Masters versorgt werden. Synchronisieren Sie alle Tabellen, und unterbinden Sie Schreibvorgänge, indem Sie eine `FLUSH TABLES WITH READ LOCK`-Anweisung auf dem Master ausführen:

```
MariaDB [(none)]> FLUSH TABLES WITH READ LOCK;
```

Listing 11.9 Tabellensynchronisierung und Stopp aller Schreibvorgänge

Wechseln Sie nun in das Datenverzeichnis des Master-Servers (`/var/lib/mysql/`). Dort gibt es für jede Datenbank ein Unterverzeichnis. Kopieren Sie die Dateien aller Datenbanken, die an der Replikation teilnehmen sollen, in das identische Verzeichnis auf dem Slave-Server. Lassen Sie dabei jedoch die Dateien `master.info` und `relay-log.info` aus.

Wenn auf dem Slave-Server andere MariaDB-Benutzerkonten als auf dem Master vorhanden sind, sollten Sie die Datenbank `mysql` besser nicht replizieren.



Während die mit `FLUSH TABLES WITH READ LOCK` erwirkte Lesesperre gültig ist, ermitteln Sie den Namen des aktuellen Binär-Logs und den Versatz auf dem Master:

```
MariaDB [(none)]> SHOW MASTER STATUS;
+-----+-----+-----+-----+
| File           | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+-----+-----+-----+-----+
| mariadb-bin.003 | 73      | test        | manual,mysql     |
+-----+-----+-----+-----+
```

Listing 11.10 Binlog-Name und Versatz auf dem Master ermitteln

Die Spalte `File` zeigt den Namen der Log-Datei an, und `Position` zeigt den Versatz innerhalb der Datei. In diesem Beispiel heißt die Binär-Log-Datei `mariadb-bin.003`, der Versatz ist 73. Notieren Sie diese Werte, denn Sie benötigen sie später beim Konfigurieren des Slaves. Die Werte stellen die Replikationskoordinaten dar, bei denen der Slave die Verarbeitung neuer Updates vom Master startet. Wenn der Master zuvor ohne aktiviertes Binär-Logging ausgeführt wurde, sind die von `SHOW MASTER STATUS` oder `mysqldump --master-data` angezeigten Werte für Log-Name und Position leer. In diesem Fall lauten die Werte, die Sie später als Log-Dateinamen und Position auf dem Slave angeben müssen, (Leer-String) und 4. Nachdem Sie die Momentaufnahme erstellt und Log-Name und Versatz aufgezeichnet haben, können Sie die Schreibaktivitäten auf dem Master neu starten:

```
MariaDB [(none)]> UNLOCK TABLES;
```

Listing 11.11 Schreibaktivitäten wieder starten

Notieren Sie den Log-Namen und Versatz aus der Ausgabe von `SHOW MASTER STATUS` (siehe Listing 11.10) wie zuvor beschrieben. Danach fahren Sie den Server herunter, ohne die Tabellen zu entsperren; hierdurch ist sichergestellt, dass der Server mit einem Status beendet wird, der den Angaben zu Log-Datei und Versatz entspricht:

```
# mysqladmin -u root shutdown
```

Listing 11.12 Den MariaDB-Server stoppen

Die Alternative: SQL-Speicherauszug

Eine Alternative, die sowohl bei MyISAM- als auch bei InnoDB-Tabellen funktioniert, besteht darin, einen SQL-Speicherauszug des Masters anstatt – wie zuvor beschrieben – einer binären Kopie zu erstellen. Hierzu führen Sie `mysqldump --master-data` auf dem Master aus und laden die SQL-Speicherauszugsdatei später in Ihren Slave. Diese Vorgehensweise ist langsamer als die Erstellung einer binären Kopie, hat aber dafür den Vorteil, dass der Slave dabei nicht gestoppt werden muss. Der Name der binären Log-Datei und der Versatz sind in diesem Fall im Dump zu finden. Die genaue Vorgehensweise finden Sie im folgenden Unterabschnitt »Konfiguration des Slave-Servers«.

Konfiguration des Slave-Servers

Um den Slave-Server zu konfigurieren, nehmen Sie auch auf ihm zunächst die Änderungen an der Datei `adminbuch.cnf` vor. Für den Slave-Betrieb genügt es, den Parameter `server_id` zu setzen:

```
[mysqld]
server_id = 101
```

Listing 11.13 Server-ID des Slaves konfigurieren

Falls Sie die Master-Master-Replikation nutzen möchten, fügen Sie an dieser Stelle in der `adminbuch.cnf` auch die weiteren Parameter hinzu:

```
log_bin                = /var/log/mysql/mariadb-bin.log
max_binlog_size        = 100M
innodb_flush_log_at_trx_commit=1
sync_binlog=1
```

Listing 11.14 Weitere Einstellungen für die Master-Master-Replikation

Unter CentOS müssen Sie auch auf dem Slave nach dem Muster aus Listing 11.8 das Verzeichnis `/var/log/mysql` anlegen und für den User `mysql` beschreibbar machen.

Start der Master-Slave-Replikation

Zunächst wird auf dem Master eine Datenbank benötigt, die repliziert werden soll. Eine Beispieldatenbank mit nur einer Tabelle und zwei Datensätzen legen Sie wie folgt an:

```
MariaDB [(none)]> create database sonnensystem;
Query OK, 1 row affected (0.01 sec)
```

```
MariaDB [(none)]> use sonnensystem;
Database changed
```

```
MariaDB [sonnensystem]> create table planeten (id integer, name varchar(20));
Query OK, 0 rows affected (0.03 sec)
```

```
MariaDB [sonnensystem]> insert into planeten values (1,'Merkur'),(2,'Venus');
Query OK, 2 rows affected (0.00 sec)
Records: 2 Duplicates: 0 Warnings: 0
```

Listing 11.15 Eine einfache Beispieldatenbank

Anschließend müssen Sie die Log-Datei und die Position in der Log-Datei ermitteln, um die Synchronisation sauber zu starten. Alle Schreibzugriffe auf die Datenbank müssen unterbunden werden (siehe Listing 11.16). Bitte beachten Sie: Sie dürfen den Client nach dem `flush tables with read lock` nicht verlassen, da sonst der Lock aufgehoben wird.

```
MariaDB [sonnensystem]> flush tables with read lock;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [sonnensystem]> show master status;
+-----+-----+-----+-----+
| File           | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+-----+-----+-----+-----+
| mariadb-bin.000001 | 1741 | | |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Listing 11.16 Schreibzugriffe unterbinden und aktuellen Status ermitteln

Diese Datenbank wird nun in einer separaten Session in eine Datei exportiert und auf den Slave-Server kopiert:

```
mysqldump sonnensystem --databases --master-data -u root -p > sonnensystem.sql
scp sonnensystem.sql slave:/tmp/
```

Listing 11.17 Die Beispieldatenbank exportieren und auf den Slave-Server kopieren

Wechseln Sie nun auf den Slave-Server, und spielen Sie die Datei in MariaDB ein:

```
mysql -u root -p < /tmp/sonnensystem.sql
```

Listing 11.18 Die Beispieldatenbank auf dem Slave-Server in MariaDB einspielen

Kontrollieren Sie anschließend am MariaDB-Prompt, ob die Datenbank vollständig und korrekt angekommen ist:

```
MariaDB [(none)]> use sonnensystem;
Reading table information for completion of table and column names
```

You can turn off this feature to get a quicker startup with -A

```
Database changed
MariaDB [sonnensystem]> select * from planeten;
+-----+-----+
| id  | name  |
+-----+-----+
|  1  | Merkur |
|  2  | Venus  |
+-----+-----+
2 rows in set (0.00 sec)
```

Listing 11.19 Die Beispieldatenbank auf dem Slave-Server überprüfen

Das hat funktioniert. Fügen Sie jetzt auf dem Master-Server einen weiteren Datensatz zur Tabelle hinzu. Dazu muss natürlich zuerst der Lock aufgehoben werden:

```
MariaDB [sonnensystem]> unlock tables;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [sonnensystem]> insert into planeten values (3,'Erde');
Query OK, 1 row affected (0.01 sec)
```

Listing 11.20 Auf dem Master einen weiteren Datensatz zur Beispieldatenbank hinzufügen

Danach starten Sie die Replikation auf dem Slave so wie in Listing 11.21 beschrieben:

```
MariaDB [sonnensystem]> show slave status;
Empty set (0.00 sec)
MariaDB [sonnensystem]> CHANGE MASTER TO MASTER_HOST='masterserver' ,
-> MASTER_USER='repl',
-> MASTER_PASSWORD='slavepass',
-> MASTER_PORT=3306,
-> MASTER_LOG_FILE='mariadb-bin.000001',
-> MASTER_LOG_POS=1741,
-> MASTER_CONNECT_RETRY=10;
```

```
MariaDB [sonnensystem]> START SLAVE;
```

Listing 11.21 Die Replikation starten

Der letzte Befehl erzeugt keine Fehlermeldungen oder Ausgaben. Untersuchen Sie jetzt noch einmal SHOW SLAVE STATUS:

```
MariaDB [sonnensystem]> SHOW SLAVE STATUS\G
***** 1. row *****
Slave_IO_State: Waiting for master to send event
Master_Host: masterserver
```



```

        Master_User: repl
        Master_Port: 3306
        Connect_Retry: 10
        Master_Log_File: mariadb-bin.000001
    Read_Master_Log_Pos: 1953
        Relay_Log_File: mariadb-relay-bin.000002
        Relay_Log_Pos: 741
    Relay_Master_Log_File: mariadb-bin.000001
        Slave_IO_Running: Yes
        Slave_SQL_Running: Yes
        Replicate_Do_DB:
    Replicate_Ignore_DB:
        Replicate_Do_Table:
    Replicate_Ignore_Table:
    Replicate_Wild_Do_Table:
    Replicate_Wild_Ignore_Table:
        Last_Errno: 0
        Last_Error:
        Skip_Counter: 0
    Exec_Master_Log_Pos: 1953
        Relay_Log_Space: 1037
        Until_Condition: None
        Until_Log_File:
        Until_Log_Pos: 0
    Master_SSL_Allowed: No
    Master_SSL_CA_File:
    Master_SSL_CA_Path:
        Master_SSL_Cert:
        Master_SSL_Cipher:
        Master_SSL_Key:
    Seconds_Behind_Master: 0
    Master_SSL_Verify_Server_Cert: No
        Last_IO_Errno: 0
        Last_IO_Error:
        Last_SQL_Errno: 0
        Last_SQL_Error:
    Replicate_Ignore_Server_Ids:
        Master_Server_Id: 10
1 row in set (0.00 sec)

```

Listing 11.22 Am MariaDB-Prompt den Slave-Status anzeigen lassen

Der gerade neu hinzugefügte Datensatz (siehe Listing 11.20) sollte nun auf den Slave repliziert worden sein. Kontrollieren Sie dies durch eine einfache `SELECT`-Abfrage auf dem Slave:

```
MariaDB [sonnensystem]> select * from planeten;
+-----+-----+
| id  | name  |
+-----+-----+
|  1  | Merkur |
|  2  | Venus  |
|  3  | Erde   |
+-----+-----+
3 rows in set (0.00 sec)
```

Listing 11.23 Prüfen, ob der neue Datensatz repliziert wurde

Es hat funktioniert: Der Datensatz mit der ID 3 ist angekommen. Wenn Sie nun einen weiteren Datensatz auf dem Master anlegen, sollte er praktisch ohne Verzögerung auf dem Slave ankommen. Fügen Sie auf dem Master noch einen vierten Datensatz hinzu:

```
MariaDB [sonnensystem]> insert into planeten values (4,'Mars');
Query OK, 1 row affected (0.00 sec)
```

Listing 11.24 Einen weiteren Datensatz auf dem Master hinzufügen

Mit der bekannten SELECT-Abfrage sollten Sie den Datensatz auf dem Slave sehen können:

```
MariaDB [sonnensystem]> select * from planeten;
+-----+-----+
| id  | name  |
+-----+-----+
|  1  | Merkur |
|  2  | Venus  |
|  3  | Erde   |
|  4  | Mars   |
+-----+-----+
4 rows in set (0.00 sec)
```

Listing 11.25 Kontrolle auf dem Slave-Server

Damit haben Sie Ihre Master-Slave-Replikation erfolgreich getestet.

11.1.3 Master-Master-Replikation

Sie können die bestehende Master-Slave-Replikation mit nur geringem Aufwand zu einer Master-Master-Replikation erweitern. Dazu legen Sie zunächst auch auf dem jetzigen Slave-Server ein Benutzerkonto für die Replikation an:

```
MariaDB [sonnensystem]> GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.*
-> TO repl@'192.168.1.%' IDENTIFIED BY 'slavepass';
```

```
### oder ###
```

```
MariaDB [sonnensystem]> GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.*
  -> TO 'repl'@'%'.mydomain.com' IDENTIFIED BY 'slavepass';
```

Listing 11.26 Einen weiteren Benutzer für die Replikation anlegen

Führen Sie nun – immer noch auf dem bisherigen Slave-Server – am MariaDB-Prompt das Kommando `SHOW MASTER STATUS` aus. Bitte setzen Sie vorher den »read lock«.

```
MariaDB [sonnensystem]> flush tables with read lock;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [sonnensystem]> show master status;
```

```
+-----+-----+-----+-----+
| File           | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+-----+-----+-----+-----+
| mariadb-bin.000002 |      98 |              |                   |
+-----+-----+-----+-----+
```

```
1 row in set (0.00 sec)
```

Listing 11.27 »show master status« auf dem bisherigen Slave-Server

Merken oder notieren Sie sich die Werte für `File` und `Position`. Sie benötigen diese Daten im nächsten Schritt.

Wechseln Sie nun auf den bisherigen Master-Server, und führen Sie am MariaDB-Prompt die Kommandos aus, die in Listing 11.28 hinter dem Prompt `MariaDB [sonnensystem]>` stehen, nachdem Sie mittels `use sonnensystem;` zur Datenbank *sonnensystem* gewechselt sind:

```
MariaDB [sonnensystem]> show slave status;
Empty set (0.00 sec)
MariaDB [sonnensystem]> change master to master_host='slaveserver',
  -> master_user='repl',
  -> master_password='slavepass',
  -> master_log_file='mariadb-bin.000002',
  -> master_log_pos=98,
  -> master_connect_retry=10;
Query OK, 0 rows affected (0.01 sec)
```

```
MariaDB [sonnensystem]> start slave;
Query OK, 0 rows affected (0.01 sec)
```

```
MariaDB [sonnensystem]> show slave status;
Slave_IO_State: Waiting for master to send event
Master_Host: slaveserver
```

```
      Master_User: repl
      Master_Port: 3306
      Connect_Retry: 60
      Master_Log_File: mariadb-bin.000002
      Read_Master_Log_Pos: 98
      Relay_Log_File: mysqld-relay-bin.000002
      Relay_Log_Pos: 235
      Relay_Master_Log_File: mariadb-bin.000002
      Slave_IO_Running: Yes
      Slave_SQL_Running: Yes
      Replicate_Do_DB:
      Replicate_Ignore_DB:
      Replicate_Do_Table:
      Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
      Replicate_Wild_Ignore_Table:
      Last_Errno: 0
      Last_Error:
      Skip_Counter: 0
      Exec_Master_Log_Pos: 98
      Relay_Log_Space: 235
      Until_Condition: None
      Until_Log_File:
      Until_Log_Pos: 0
      Master_SSL_Allowed: No
      Master_SSL_CA_File:
      Master_SSL_CA_Path:
      Master_SSL_Cert:
      Master_SSL_Cipher:
      Master_SSL_Key:
      Seconds_Behind_Master: 0
1 row in set (0.00 sec)
```

Listing 11.28 Den bisherigen Slave zum zweiten Master-Server befördern

Das letzte SELECT-Statement in Listing 11.29 zeigt den Inhalt der Tabelle Planeten auf dem bisherigen Master-Server:

```
MariaDB [sonnensystem]> unlock tables;
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [sonnensystem]> insert into planeten values (5,'Jupiter'),(6,'Saturn');
Query OK, 2 rows affected (0.01 sec)
Records: 2 Duplicates: 0 Warnings: 0
```

```
MariaDB [sonnensystem]> select * from planeten;
+-----+-----+
| id  | name  |
+-----+-----+
|  1  | Merkur |
|  2  | Venus  |
|  3  | Erde   |
|  4  | Mars   |
|  5  | Jupiter |
|  6  | Saturn |
+-----+-----+
6 rows in set (0.00 sec)
```

Listing 11.29 Zwei weitere Planeten anlegen

Wenn Sie nun auf dem ehemaligen Slave-Server einen Datensatz hinzufügen, wird er auf den Master repliziert werden. Wechseln Sie also auf den Ex-Slave-Server, und fügen Sie der Tabelle einen weiteren Planeten hinzu:

```
MariaDB [sonnensystem]> insert into planeten values (7,'Uranus');
Query OK, 1 row affected (0.01 sec)
```

Listing 11.30 Einen Datensatz auf dem ehemaligen Slave-Server hinzufügen

Wechseln Sie nun zurück auf den Master-Server, und kontrollieren Sie, ob der Datensatz auch dorthin repliziert wurde (siehe Listing 11.31). Sollte das nicht der Fall sein, gehen Sie bitte die vorhergehenden Schritte noch einmal durch.

```
MariaDB [sonnensystem]> select * from planeten;
+-----+-----+
| id  | name  |
+-----+-----+
|  1  | Merkur |
|  2  | Venus  |
|  3  | Erde   |
|  4  | Mars   |
|  5  | Jupiter |
|  6  | Saturn |
|  7  | Uranus |
+-----+-----+
7 rows in set (0.00 sec)
```

Listing 11.31 Erfolgskontrolle auf dem ehemaligen Master-Server

MariaDB erlaubt explizit die Ring-Replikation (*Multi-Master-Replikation*). Damit können Sie mehr als zwei Server zu einem Ring zusammenfassen. Wenn der Ring unterbro-



chen ist, ist an der Stelle auch die Replikation für alle beteiligten Server unterbrochen. Hier sind besondere Vorkehrungen zu treffen. Weitere Informationen dazu finden Sie auf den Webseiten, die von der folgenden URL aus verlinkt sind: <https://mariadb.com/kb/en/library/multi-source-replication>

11.2 Tuning

Beim Tuning eines MariaDB-Servers ist es das zentrale Ziel, die Festplattenlast, also die Anzahl der Ein-/Ausgabeoperationen (I/O), zu verringern. Während ein RAM-Zugriff in wenigen Nanosekunden erfolgt, bewegt man sich bei Festplattenzugriffen bereits im Millisekundenbereich, weil das Positionieren der Schreib-/Leseköpfe nun einmal diese Zeit in Anspruch nimmt. Auch die Drehgeschwindigkeit der Festplattenscheiben kann nicht beliebig gesteigert werden. Das vorher Gesagte spielt zwar im Zeitalter von SSDs und NVMe eine untergeordnete Rolle, ist aber trotzdem nicht zu vernachlässigen.

Ferner sollten Sie auf die Latenz und die Geschwindigkeit der Netzwerkanbindung achten, denn wenn der Server seine Anfragen zu langsam erhält oder seine Antworten nicht schnell genug loswird, geht wertvolle Zeit verloren. Sollten Sie mit dem Gedanken spielen, Ihren Datenbankserver mit besserer Hardware zu bestücken, so sollten Sie zunächst in RAM investieren. Schnellere CPUs bringen nur bei komplexen Rechenoperationen einen Vorteil. Der Umstieg von lokalen Festplatten auf SSDs, NVMe oder ein schnelles, gut angebundenes SAN beschleunigt das I/O-Verhalten. In den folgenden Abschnitten geht es aber nicht um Hardware-Tuning.

Stattdessen werden Sie lernen, wie Sie mit den Bordmitteln des Betriebssystems und MariaDB die Speichernutzung und das Laufzeitverhalten Ihres Servers optimieren können.



Nutzen Sie »mytop«

Das Programm *mytop* zeigt Ihnen eine ständig aktualisierte Übersicht darüber, womit Ihr MariaDB-Server gerade beschäftigt ist. Es liefert auch eine Reihe grundlegender Messwerte, an denen Sie ablesen können, ob Ihre Tuning-Maßnahmen erfolgreich sind.

11.2.1 Tuning des Speichers

Unter der URL <https://launchpad.net/mysql-tuning-primer> erhalten Sie das sehr nützliche *bash*-Skript *mysql-tuning-primer.sh*. Wenn Sie Debian oder Ubuntu nutzen, ändern Sie am Anfang des Skripts `/bin/sh` in `/bin/bash`. Das ist notwendig, weil `/bin/sh` unter Ubuntu ein Link auf `/bin/dash` ist. Beachten Sie bitte auch, dass das Skript das Programm *bc* erfordert. Bitte installieren Sie es, falls es noch nicht vorhanden ist. Führen Sie das Skript auf dem Server aus, den Sie tunen möchten. Beachten Sie dabei die beiden folgenden Punkte:

- ▶ Der MariaDB-Server sollte nicht brandneu, sondern schon ein wenig »eingefahren« sein, sonst liefern die Analysen der Speicher- und Cache-Nutzung irreführende Ergebnisse.
- ▶ Der Server sollte mindestens 48 Stunden laufen, bevor das Skript gestartet wird, damit es einen guten Überblick über die typische Lastsituation auf dem Server bekommt. Das Skript sollte außerdem zu den normalen Betriebszeiten laufen. Wenn der Server beim Start des Skripts noch keine 48 Stunden Uptime aufweist, erhalten Sie folgende Warnung:
Warning: Server has not been running for at least 48hrs.
It may not be safe to use these recommendations

Listing 11.32 Warnung vor zu kurzer Laufzeit des Systems

Auf den folgenden Seiten finden Sie die für Ihre Analysen relevanten Ausgaben des Skripts *mysql-tuning-primer.sh* und Hinweise darauf, wie Sie die Ausgaben interpretieren und als Grundlage für Tuning-Maßnahmen nutzen können.

11

Slow Queries

SLOW QUERIES

The slow query log is NOT enabled.

Current long_query_time = 10.000000 sec.

You have 0 out of 1908 that take longer than 10.000000 sec. to complete

Your long_query_time seems to be fine

Listing 11.33 Skriptausgabe: »Slow Queries«

Die Behandlung von *Slow Queries*, also von langsamen Abfragen, wird in Abschnitt 11.2.2, »Tuning von Indizes«, näher betrachtet.

Binary Update Log

BINARY UPDATE LOG

The binary update log is NOT enabled.

You will not be able to do point in time recovery

See <http://dev.mysql.com/doc/refman/5.1/en/point-in-time-recovery.html>

Listing 11.34 Skriptausgabe: »Binary Update Log«

Auch das *Binary Update Log* – die *binäre Logdatei* – begegnet uns erst an späterer Stelle wieder, und zwar in Abschnitt 11.3, »Backup und Point-In-Time-Recovery«.

Max Connections

MAX CONNECTIONS

Current max_connections = 151

Current threads_connected = 1

Historic max_used_connections = 1

The number of used connections is 0% of the configured maximum.

You are using less than 10% of your configured `max_connections`.
Lowering `max_connections` could help to avoid an over-allocation of memory
See "MEMORY USAGE" section to make sure you are not over-allocating

Listing 11.35 Skriptaussgabe: »Max Connections«

Jede Verbindung benötigt 2 MB Speicher. Dazu kommen aber zusätzlich noch die Puffer und Caches, deren Größe in den Variablen `sort_buffer_size`, `read_buffer_size` und `binlog_cache_size` definiert ist.

Das Skript hat festgestellt, dass wesentlich weniger Verbindungen benutzt werden, als im Parameter `max_connections` konfiguriert sind. Da für jede potenzielle Verbindung Speicher reserviert wird, ist es sinnvoll, den `max_connections`-Wert abzusenken. In der MariaDB-Shell geht das wie folgt:

```
MariaDB [(none)]> set global max_connections=10;
Query OK, 0 rows affected (0.00 sec)
MariaDB [(none)]> show variables like 'max_connections';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| max_connections | 10    |
+-----+-----+
1 row in set (0.00 sec)
```

Listing 11.36 Der Wert »`max_connections`« wird zunächst reduziert, dann kontrolliert.

Die Einstellung des `max_connections`-Werts wird auf diese Weise nur temporär verändert. Wenn Sie den Wert dauerhaft verändern wollen, gehen Sie wie folgt vor:

- ▶ **Debian/Ubuntu:** Legen Sie eine neue Datei im Verzeichnis `/etc/mysql/conf.d/` an, deren Name auf `.cnf` enden muss, beispielsweise `/etc/mysql/conf.d/max_connections.cnf`. In die Datei schreiben Sie den *Section Header* `[mysqld]` und den gewünschten neuen Wert der Variablen `max_connections`:

```
[mysqld]
max_connections=10
```

Listing 11.37 Inhalt der Datei »`/etc/mysql/conf.d/max_connections.cnf`«

- ▶ **CentOS/openSUSE:** Speichern Sie die Datei aus Listing 11.37 unter `/etc/my.cnf.d` ab.

InnoDB Status

```
INNODB STATUS
Current InnoDB index space = 0 bytes
Current InnoDB data space = 0 bytes
```



```
Current InnoDB buffer pool free = 96 %
Current innodb_buffer_pool_size = 8 M
Depending on how much space your innodb indexes take up it may be safe
to increase this value to up to 2 / 3 of total system memory
```

Listing 11.38 Skriptaussgabe: »InnoDB Status«

Die verwendete Größe der *InnoDB-Buffer-Pools* können Sie über den Wert der Variablen `innodb_buffer_pool_size` anpassen. Das Vorgehen ist das gleiche wie bei der Variablen `max_connections`. In diesem Beispiel ist keine Änderung notwendig.

Memory Usage

```
MEMORY USAGE
Max Memory Ever Allocated : 44 M
Configured Max Per-thread Buffers : 405 M
Configured Max Global Buffers : 42 M
Configured Max Memory Limit : 447 M
Physical Memory : 244 M
Max memory limit exceeds 90% of physical memory
```

Listing 11.39 Skriptaussgabe: »Memory Usage«

Diesen Wert müssen Sie unbedingt im Auge behalten! Sollte das konfigurierte *Memory Limit* über dem tatsächlich verfügbaren Speicher liegen, kann es zum *Swappen* (zur Nutzung von Auslagerungsspeicher) kommen. Der Server würde in diesem Fall sehr langsam werden.



Key Buffer

```
KEY BUFFER
Current MyISAM index space = 301 M
Current key_buffer_size = 64 M
Key cache miss rate is 1 : 156
Key buffer free ratio = 40 %
Your key_buffer_size seems to be too high.
Perhaps you can use these resources elsewhere
```

Listing 11.40 Skriptaussgabe: »Key Buffer«

Die Indexblöcke aller *MyISAM*-Tabellen werden in einen Pufferspeicher² geschrieben, der von allen Threads gemeinsam genutzt wird. Die Variable `key_buffer_size` enthält die Größe dieses Pufferspeichers. In dem Beispiel aus Listing 11.40 ist er ein wenig zu groß und könnte gefahrlos reduziert werden.

² Dieser Pufferspeicher für die *MyISAM*-Indexblöcke wird auch *Schlüssel-Cache* genannt.

Query Cache

```
QUERY CACHE
Query cache is enabled
Current query_cache_size = 128 M
Current query_cache_used = 74 M
Current query_cache_limit = 4 M
Current Query cache Memory fill ratio = 57.94 %
Current query_cache_min_res_unit = 4 K
MariaDB won't cache query results that are larger than query_cache_limit in size
```

Listing 11.41 Skriptausgabe: »Query Cache«

Die Variable `query_cache_size` definiert die Größe des Caches, der zum Zwischenspeichern von Abfrageergebnissen insgesamt zur Verfügung steht. Der Standardwert ist 0 (Null), das bedeutet, dass der Cache deaktiviert ist. Im Beispiel wurden dem Cache 128 MB zur Verfügung gestellt. Es werden nur die Ergebnisse zwischengespeichert, die nicht größer sind als der Wert in `query_cache_limit`. In diesem Beispiel liegt das Limit bei 4 MB, der Default-Wert ist 1 MB.

Sort Operations

```
SORT OPERATIONS
Current sort_buffer_size = 8 M
Current read_rnd_buffer_size = 256 K
Sort buffer seems to be fine
```

Listing 11.42 Skriptausgabe: »Sort Operations«

Der Sortierungspuffer (*Sort Buffer*) wird beispielsweise für Abfragen benötigt, die ein ORDER BY-Statement enthalten. Je nach Anzahl der Datensätze, die regelmäßig sortiert werden müssen, kann dieser Wert größer oder kleiner gewählt werden.

Joins

```
JOINS
Current join_buffer_size = 2.00 M
You have had 1138756 queries where a join could not use an index properly
You should enable "log-queries-not-using-indexes"
Then look for non indexed joins in the slow query log.
```

If you are unable to optimize your queries you may want to increase your `join_buffer_size` to accommodate larger joins in one pass.
Note! This script will still suggest raising the `join_buffer_size` when ANY joins not using indexes are found.

Listing 11.43 Skriptausgabe: »Joins«

Ein Beispiel dieses Werts wird in Abschnitt 11.2.2, »Tuning von Indizes«, näher beleuchtet.

Open Files Limit

```
OPEN FILES LIMIT
Current open_files_limit = 4096 files
```

The `open_files_limit` should typically be set to at least 2x-3x that of `table_cache` if you have heavy MyISAM usage. Your `open_files_limit` value seems to be fine

Listing 11.44 Skriptausgabe: »Open Files Limit«

Dies ist kein MariaDB-Parameter: Für die Limitierung der gleichzeitig geöffneten Dateien ist das Betriebssystem zuständig. Um diesen Wert auf 4096 festzulegen, geben Sie vor dem Start von MariaDB als *root* das Kommando `ulimit -n 4096` auf der Kommandozeile ein.

Table Cache

```
Current table_cache value = 1993 tables
You have a total of 780 tables
```

You have 1582 open tables.
The `table_cache` value seems to be fine

Listing 11.45 Skriptausgabe: »Table Cache«

Dies ist die Anzahl der geöffneten Tabellen für alle Threads. Wenn Sie nun den Wert für `table_cache` erhöhen, benötigt MariaDB mehr Dateideskriptoren. Durch die Eingabe des Befehls `flush tables` in der MariaDB-Shell werden alle offenen Tabellen geschlossen und nur diejenigen wieder geöffnet, die tatsächlich benötigt werden.

Temp Tables

```
TEMP TABLES
Current max_heap_table_size = 32 M
Current tmp_table_size = 32 M
```

Of 744743 temp tables, 15% were created on disk
Created disk tmp tables ratio seems fine

Listing 11.46 Skriptausgabe: »Temp Tables«

Wenn Sie viele Abfragen mit `GROUP BY`-Statements haben, ist es vorteilhaft, diese Werte (im Beispiel 32 MB) zu erhöhen, sofern Sie ausreichend RAM zur Verfügung haben.

Wächst die Größe der temporären Tabelle im Arbeitsspeicher über diese Beschränkung hinaus, wird MariaDB sie automatisch in eine *MyISAM*-Tabelle auf der Festplatte umwandeln, was sich natürlich nachteilig auf die Zugriffsgeschwindigkeit auswirken wird.

Table Scans

```
TABLE SCANS
Current read_buffer_size = 508 K
Current table scan ratio = 2313 : 1
read_buffer_size seems to be fine
```

Listing 11.47 Skriptausgabe: »Table Scans«

Ein *Table Scan* ist unvermeidlich, wenn eine Tabelle keinen Index besitzt. Manchmal führt MariaDB trotz des vorhandenen Index einen *Table Scan* aus. Das passiert, wenn der eingebaute Optimierer der Ansicht ist, es sei schneller, die Tabelle komplett zu lesen, als nur einzelne Datensätze herauszusuchen.

Table Locking

```
TABLE LOCKING
Current Lock Wait ratio = 1 : 1198
```

```
You may benefit from selective use of InnoDB.
If you have long running SELECT's against MyISAM tables and perform
frequent updates consider setting 'low_priority_updates=1'
If you have a high concurrency of inserts on Dynamic row-length tables
consider setting 'concurrent_insert=2'.
```

Listing 11.48 Skriptausgabe: »Table Locking«

Wenn Sie *MyISAM*-Tabellen benutzen, werden Sie sehr wahrscheinlich davon profitieren, den Parameter `concurrent_insert=2` zu aktivieren. MariaDB erlaubt damit die gleichzeitige Ausführung von `SELECT`- und `INSERT`-Statements. Nehmen Sie sich die Zeit, die für Sie beste Kombination dieser Parameter herauszufinden. Da es kein Patentrezept gibt, werden Sie um eigene Tests nicht herumkommen. Die Mühe lohnt sich, denn mit den richtigen Einstellungen holen Sie das Maximum aus Ihrem Datenbankserver heraus.

11.2.2 Tuning von Indizes

Im vorigen Abschnitt wurden *Slow Queries* bereits ebenso erwähnt wie Abfragen, die keine Indizes benutzen. MariaDB kann alle Abfragen, die länger als eine bestimmte Zeit benötigen, in eine darauf spezialisierte Log-Datei schreiben, in das *Slow Query Log*. Aktiviert wird es durch den folgenden Eintrag in der MariaDB-Konfigurationsdatei:

```
log_slow_queries = /var/log/mysql/mysql-slow.log
long_query_time = 2
log-queries-not-using-indexes
```

Listing 11.49 Aktivierung des »Slow Query Log«

In diesem Beispiel wandern alle Abfragen, die länger als zwei Sekunden brauchen, in die Log-Datei `/var/log/mysql/mysql-slow.log`. Abfragen, die keinen Index benutzen, werden dort ebenfalls protokolliert (`log-queries-not-using-indexes`).

Passen Sie »long_query_time« an

Ob der Schwellenwert von zwei Sekunden für Ihren Datenbankserver vernünftig gewählt ist, ist keineswegs sicher. Wenn Sie ein *Data Warehouse* betreiben, sind lange Abfragezeiten normal. Wenn Sie dagegen eine Webseite oder einen Online-Shop betreiben, kommt es eher auf schnelle Antwortzeiten an.

Wenn Sie einen hohen Prozentsatz von langsamen Abfragen im *Slow Query Log* finden, lohnt es sich, diesen mit dem SQL-Kommando `EXPLAIN` auf den Grund zu gehen. Das Erzeugen zusätzlicher oder besserer Indizes kann die `SELECT`-Performance drastisch erhöhen, allerdings dauern damit auch `INSERT`- und `UPDATE`-Statements deutlich länger. Seien Sie in jedem Fall nicht allzu pedantisch. Man muss nicht jede einzelne Abfrage optimieren. Solange es nicht überhandnimmt, dürfen einzelne Abfragen auch einmal etwas länger dauern – in einer Datenbank, in der es sehr häufig Aktualisierungen gibt, ist es vielleicht besser, auf Abfragen zu warten, als die Aktualisierungen warten zu lassen. Was generell besser ist, hängt von Ihrer speziellen Anwendung ab.

Ein Beispiel aus der Praxis

Das *Slow Query Log* können Sie mit dem folgenden Befehl zusammenfassen lassen:

```
mysqldumpslow /var/log/mysql/mysql-slow.log
```

Listing 11.50 Eine Zusammenfassung des »Slow Query Log« erzeugen

Wenn Sie als Parameter noch ein `-r` eingeben, können Sie sich die am längsten laufenden Abfragen zum Schluss ausgeben lassen. Achtung: Damit finden Sie nur den Typ der Abfrage, Werte werden durch den Platzhalter »N« ersetzt.



Wenn Sie nun im *slow-query.log* nach einer Abfrage dieses Typs suchen, können Sie sich diese mit dem Befehl `EXPLAIN` erklären lassen. So lässt sich beispielsweise in der Datenbank eines Mailservers folgende Abfrage definieren:

```
MariaDB [mailserver]> select * from virtual_aliases where \
    destination='mail@example.com';
```

Listing 11.51 Eine einfache Datenbankabfrage

Mit `EXPLAIN` können Sie sehen, wie der MariaDB-Server die Abfrage bearbeitet:

```
MariaDB [mailserver]> explain select * from virtual_aliases where \
    destination='mail@example.com';
```

```

+-----+-----+-----+-----+-----+-----+
| id | select_type | table          | type | possible_keys | key  | key_len |
+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE      | virtual_aliases | ALL  | NULL          | NULL | NULL    |
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+
| ref | rows | Extra          |
+-----+-----+-----+
| NULL | 127 | Using where    |
+-----+-----+-----+
1 row in set (0.00 sec)

```

Listing 11.52 Anwendung des »explain«-Statements

Schauen wir uns nun die einzelnen Elemente der Ausgabe an:

- ▶ **id**
Die *id* gibt die Sequenznummer der SELECT-Anweisung innerhalb der Abfrage an.
- ▶ **select_type**
Tabelle 11.1 zeigt die möglichen SELECT-Typen:

Typ	Bedeutung
PRIMARY	äußerste SELECT-Anweisung
SIMPLE	einfache SELECT-Anweisung (ohne UNION oder Unterabfragen)
DERIVED	abgeleitete Tabellen-SELECT-Anweisung (Unterabfrage in FROM-Klausel)
SUBQUERY	erste SELECT-Anweisung in einer Unterabfrage
DEPENDENT SUBQUERY	erste SELECT-Anweisung in einer Unterabfrage, abhängig von der äußeren Abfrage
UNCACHABLE SUBQUERY	erste SELECT-Anweisung in einer Unterabfrage, die nicht gecacht werden kann
UNION	zweite oder spätere SELECT-Anweisung in einer UNION
UNION RESULT	Ergebnis einer UNION
DEPENDENT UNION	zweite oder spätere SELECT-Anweisung in einer UNION, abhängig von der äußeren Abfrage
UNCACHABLE UNION	nicht cachbare UNION

Tabelle 11.1 »SELECT«-Typen

- ▶ `table`
Dies ist der Name der Tabelle, auf die sich die Abfrage bezieht.
- ▶ `type`
Hier wird der Join-Typ angegeben. ALL bedeutet, dass ein kompletter Tablescan durchgeführt wird. Das kann bei großen Tabellen sehr lange dauern, es besteht Handlungsbedarf. INDEX hätte angegeben, dass der Indexbaum gescannt worden ist. Weitere Möglichkeiten wären (von der schlechtesten zur besten): `range`, `index_subquery`, `unique_subquery`, `index_merge`, `ref_or_null`, `ref`, `eq_ref`, `const`, `system`. Die Erläuterungen hierzu entnehmen Sie bitte der Webseite zu EXPLAIN unter der Adresse <https://mariadb.com/kb/en/explain/>.
- ▶ `possible_keys`
Dieser Wert gibt an, welche Keys (Indizes) möglich wären; in unserem Fall keiner.
- ▶ `key`
Das ist der Index, der letztendlich benutzt wurde.
- ▶ `key_len`
Der Wert zeigt die Länge des Schlüssels an.
- ▶ `ref`
Die Referenz, die als Schlüsselwert verwendet wird.
- ▶ `rows`
Dies ist die Anzahl der Zeilen, die MariaDB in seine Untersuchung mit einbezieht.
- ▶ `extra`
Hier zeigt MariaDB an, wie es die Abfrage auflöst.

Für unser Beispiel ist die Erkenntnis wichtig, dass es keinen passenden Index gibt. Mit dem folgenden Kommando am MariaDB-Prompt erzeugen wir ihn:

```
MariaDB [(none)]> create index idx_destination on virtual_aliases(destination);
Query OK, 127 rows affected (0.12 sec)
Records: 127 Duplicates: 0 Warnings: 0
```

Listing 11.53 Nachträglich einen Index erzeugen

Mit einem weiteren EXPLAIN-Statement kontrollieren wir den Erfolg:

```
MariaDB [(none)]> explain select * from virtual_aliases where \
    destination='mail@example.com';
```

```
+----+-----+-----+-----+-----+
| id | select_type | table          | type | possible_keys | key
+----+-----+-----+-----+-----+
| 1 | SIMPLE      | virtual_aliases | ref  | idx_destination | idx_destination
+----+-----+-----+-----+-----+
```

```

+-----+-----+-----+-----+
| key_len | ref  | rows | Extra      |
+-----+-----+-----+-----+
| 82      | const | 1    | Using where |
+-----+-----+-----+-----+

```

Listing 11.54 Erfolgskontrolle der Indexerstellung

Dieser Index würde Abfragen auf die Datenbank in Zukunft deutlich beschleunigen.

Zusammenfassung der Indexerstellung

1. Schalten Sie das *Slow Query Log* ein.
2. Finden Sie mit `mysqldumpslow` die Abfrageklasse, die das System am stärksten beeinflusst.
3. Suchen Sie aus dem *Slow Query Log* ein typisches Beispiel dafür.
4. Untersuchen Sie dieses Beispiel mithilfe des `EXPLAIN`-Kommandos.
5. Erzeugen Sie einen (zusätzlichen) Index.

11.3 Backup und Point-In-Time-Recovery

Sicherlich legen Sie in bestimmten Abständen ein vollständiges Backup Ihrer Datenbanken an. Sollte es zu einem Datenverlust kommen, können Sie das Backup hoffentlich wieder einspielen (*Restore*). Änderungen, die seit dem Anlegen des Backups vorgenommen wurden, sind aber leider verloren – es sei denn, Sie nutzen *Point-In-Time-Recovery*. *Point-In-Time-Recovery* bedeutet, dass Sie die Änderungen am Datenbestand, die seit einem bestimmten Zeitpunkt in der Vergangenheit vorgenommen worden sind, ebenfalls restaurieren können. Das Mittel, das Ihnen diesen Kunstgriff ermöglicht, ist das *binäre Logging*. Dabei werden die Änderungen inkrementell in Binärdateien geschrieben, aus denen sie im Bedarfsfall wiederhergestellt werden können. Damit Sie *Point-In-Time-Recovery* nutzen können, müssen Sie zunächst dafür sorgen, dass Ihr Server die binären Logfiles schreibt. Fügen Sie der MariaDB-Konfiguration dazu folgende Zeilen hinzu (bitte denken Sie unter CentOS daran, das Verzeichnis zu erstellen und zu berechtigen; sehen Sie sich dazu auch Listing 11.8 an):

```

log_bin                = /var/log/mysql/mariadb-bin.log
max_binlog_size        = 100M
innodb_flush_log_at_trx_commit=1
sync_binlog=1

```

Listing 11.55 Binäres Logging aktivieren

Nach einem Neustart des Datenbanksservers werden die gesamten binären Log-Dateien unter `/var/log/mysql/` angelegt. Für die folgenden Beispiele nutzen wir eine Datenbank mit dem Namen *sonnensystem*.

11.3.1 Restore zum letztmöglichen Zeitpunkt

Erstellen Sie ein vollständiges Backup der Datenbank mit diesem Kommando:

```
mysqldump --flush-logs --single-transaction sonnensystem -u root -p > backup.sql
```

Listing 11.56 Vollständiges Backup einer Datenbank

Wenn nun zu einem späteren Zeitpunkt – das heißt, nachdem seit dem Backup weitere Daten in der Datenbank geändert wurden – eine Katastrophe passiert und ein Restore notwendig wird, spielen Sie zunächst das Backup wieder ein:

```
mysql -u root -p < backup.sql
```

Listing 11.57 Restore der gesicherten Datenbank

Nun müssen noch die Änderungen restauriert werden, die seit dem Erstellen des Backups vorgenommen wurden. Diese Änderungen finden sich in den binären Logfiles. Finden Sie zunächst heraus, wie viele binäre Logfiles seit dem Erstellen des Backups angelegt wurden. Sollten es mehrere sein, müssen Sie zum Wiederherstellen alle Logfiles angeben. Listing 11.58 zeigt ein Restore aus zwei binären Logfiles (es handelt sich um eine einzige Zeile, die nur aufgrund der Seitenbreite umbrochen wurde):

```
mysqlbinlog --database sonnensystem /var/log/mysql/mariadb-bin.000001 \
/var/log/mysql/mariadb-bin.000002 | mysql -u root -p
```

Listing 11.58 Restore aus den binären Logfiles

11.3.2 Restore zu einem bestimmten Zeitpunkt

Wenn Sie einen falschen Befehl eingegeben haben, wie beispielsweise das versehentliche Löschen einer Tabelle, können Sie per *Point-In-Time-Recovery* auch ein Restore genau bis zu dem Zeitpunkt durchführen, bevor Sie die Tabelle gelöscht haben. Nehmen wir zum Beispiel an, die Tabelle *planeten* in der Datenbank *sonnensystem* ist mit dem Befehl `drop table planeten` gelöscht worden. Der Fehler ist Ihnen glücklicherweise sofort aufgefallen. Suchen Sie im Verzeichnis `/var/log/mysql/` das aktuellste der dort abgelegten binären Logfiles (Sie benötigen das aktuellste, weil der Fehler ja gerade erst passiert ist). In unserer Musterdatenbank hat das Verzeichnis beispielsweise den in Listing 11.59 gezeigten Inhalt:

```
# ls -l /var/log/mysql
total 60
-rw-rw---- 1 mysql mysql  353 28. Aug 17:17 mariadb-bin.000001
-rw-rw---- 1 mysql mysql  353 28. Aug 17:20 mariadb-bin.000002
-rw-rw---- 1 mysql mysql  353  6. Okt 09:29 mariadb-bin.000003
-rw-rw---- 1 mysql mysql  136  6. Okt 10:22 mariadb-bin.index
-rw-rw---- 1 mysql mysql 33085  6. Okt 11:44 mysqld.log
```

Listing 11.59 Inhalt des Verzeichnisses `»/var/log/mysql/«`

Das aktuellste binäre Logfile ist hier die Datei *mariadb-bin.000003*. Führen Sie auf der Kommandozeile den folgenden Befehl aus:

```
mysqlbinlog ---database sonnensystem /var/log/mysql/mariadb-bin.000003
```

Listing 11.60 Den Inhalt des binären Logfiles anschauen

Sie erhalten unter anderem die Ausgabe aus Listing 11.61. Dort sehen Sie, dass an Position 1897 das Löschen der Tabelle gestartet wurde (at 1897) und dass es an Position 2019 beendet wurde (end_log_pos 2019):

```
# at 1855
#181006 12:06:40 server id 105  end_log_pos 1897 CRC32 0xac3f2172      \
GTID 0-105-11 ddl
/*!100001 SET @@session.gtid_seq_no=11*//*!*/;
# at 1897
#181006 12:06:40 server id 105  end_log_pos 2019 CRC32 0x60355708      \
Query  thread_id=30      exec_time=0      error_code=0
SET TIMESTAMP=1538820400*//*!*/;
DROP TABLE `planeten` /* generated by server */
*//*!*/;
DELIMITER ;
# End of log file
ROLLBACK /* added by mysqlbinlog */;
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
```

Listing 11.61 Ergebnis der Suche im binären Logfile (Auszug)

Nun spielen Sie das letzte vollständige Backup wieder ein:

```
mysql -u root -p < backup.sql
```

Listing 11.62 Restore der gesicherten Datenbank

Als Nächstes restaurieren Sie alle Änderungen bis zum fatalen DROP TABLE:

```
mysqlbinlog --database=sonnensystem /var/log/mysql/mariadb-bin.000003 \
--stop-position=1897 | mysql -u root -p
```

Listing 11.63 Restore der Daten bis zum »Drop Table«

Falls Sie nach dem DROP TABLE noch weitergearbeitet haben, können Sie auch diese Änderungen noch zurückholen, indem Sie auch die Daten nach dem Zeitstempel 1897 zurückspielen:

```
mysqlbinlog --database=sonnensystem /var/log/mysql/mariadb-bin.000007 \
--start-position=2019 | mysql -u root -p
```

Listing 11.64 Restore der Daten ab dem Zeitstempel »2019«

Kapitel 12

Syslog

In diesem Kapitel geht es darum, wie Sie den Zustand Ihres Systems überprüfen und eventuelle Probleme besser lösen können. Denn das ist eine der Hauptaufgaben bei der Systemadministration. Für diese Aufgabe sollten Sie alle Möglichkeiten nutzen, vor allen Dingen die, die Ihnen das System ohne weitere Programme installieren und gar kaufen zu müssen.

Das Syslog-Protokoll und seine erste Implementation stammen aus der BSD-Welt. Eric Allmann entwickelte sie als Logging-Instanz für Sendmail. Syslog wurde erst 2001, viele Jahre nach seiner Entstehung, in den RFCs 3164 und 3195 dokumentiert, und auch diese Standarddokumente lassen noch reichlich Spielraum für die Kreativität der Entwickler. Aus diesem Grund gibt es heute mehrere Syslog-Implementationen, die in ihrer grundlegenden Funktionalität miteinander kompatibel sind, aber auch Besonderheiten und oft nützliche Erweiterungen enthalten. Das Syslog-Protokoll benutzt im Netzwerk UDP auf Port 514. Aktuellere Syslog-Implementationen, die im weiteren Verlauf dieses Kapitels beschrieben werden, beherrschen auch den Transport per TCP.

Fast alle Distributionen setzen heute auf den `systemd` zusammen mit dem `journald`, so auch alle Distributionen die wir hier im Buch ansprechen. Der `journald` zusammen mit dem Kommando `journalctl` ist die aktuellste Art und Weise wie Sie die Log-Meldungen Ihres Systems auswerten können. Trotzdem werden wir hier auch noch einen Blick auf die älteren Syslog-Dienste werfen, da Sie bei einigen Distributionen immer noch bereitgestellt werden. Auf die Dauer werden alle Distributionen jedoch ganz oder zumindest zum größten Teil auf den `systemd` umstellen. Dann wird dieses Wissen noch wichtiger.

12.1 Der Aufbau von Syslog-Nachrichten

Alle Prozesse generieren mehr oder weniger Meldungen zu ihrem Zustand, bestimmten Aktionen oder aufgetretenen Fehlern. Jede Nachricht wird dann, abhängig von dem verwendeten Log-System, in eine Datei oder wie im Falle von `systemd` in eine Binärdatei geschrieben. Eine Syslog-Nachricht besteht dabei aus drei Teilen. Dies sind:

- ▶ der **Priority-Selektor**: Dieser nur ein Byte große Selektor enthält zwei Felder; beide kodieren einen Zahlwert. Das erste Feld enthält die sogenannte *Facility* (etwa: Quelle, Herkunft), das zweite die *Severity* (»Log-Level«, etwa: Wichtigkeit). Die *Facility* gibt an, wo der

Schuh drückt, und die *Severity* beschreibt, wie sehr es schmerzt. Allein der Priority-Selektor gibt dem Admin also einen ersten Eindruck von der Situation: Bekommt er nur eine belanglose Information, oder steht das System kurz vor der Kernschmelze?

- ▶ der **Header**: Er enthält die IP-Adresse des absendenden Systems und einen Zeitstempel. Der Zeitstempel wird vom empfangenden Syslog-Daemon gesetzt, nicht vom Absender.
- ▶ die **Nachricht**: Hier steht die eigentliche Meldung. Das Syslog-Protokoll sieht dafür eine Länge von maximal 1.024 Bytes vor.

In Tabelle 12.1 sehen Sie die wichtigsten Syslog-Quellen und deren Bedeutung.

Facility	Bedeutung
LOG_KERN	Kernelmeldungen
LOG_USER	Meldungen von Userspace-Programmen
LOG_MAIL	Mail-Subsystem
LOG_DAEMON	Serverdienste
LOG_SYSLOG	Syslog-interne Meldungen
LOG_LPR	Druckersubsystem
LOG_NEWS	Usenet-Server
LOG_CRON	Meldungen des Cron-Daemons
LOG_AUTHPRIV	Login- und Autorisierungsmeldungen
LOG_LOCAL0 – LOG_LOCAL7	reserviert für lokale Nutzung

Tabelle 12.1 Die wichtigsten Syslog-Quellen (Facilitys)

In Tabelle 12.2 sind die acht Log-Level des Syslog und deren Bedeutung aufgeführt.

Level	Bedeutung
LOG_EMERG	Das System ist unbenutzbar.
LOG_ALERT	Alarm, schnelles Handeln erforderlich
LOG_CRIT	kritischer Fehler
LOG_ERR	normaler Fehler

Tabelle 12.2 Log-Level

Level	Bedeutung
LOG_WARNING	Warnung
LOG_NOTICE	wichtige Benachrichtigung
LOG_INFO	Benachrichtigung
LOG_DEBUG	Informationen für Entwickler, Hilfe bei der Fehlersuche

Tabelle 12.2 Log-Level (Forts.)

Alle im Buch vorgestellten Distributionen (Debian, Ubuntu, CentOS und openSUSE Leap) verwenden systemd zusammen mit dem *journal*d. Wollen Sie eines der anderen hier im Buch beschriebenen Log-Systeme verwenden, müssen Sie den Dienst immer installieren.

12.2 systemd mit journalctl

Beginnen wollen wir mit dem systemd zusammen mit den Kommando `journalctl`, da das bei allen Distributionen aktuell und in der Zukunft der Standard sein wird. Alle Alternativen, die Ihnen zur Verfügung stehen, benötigen immer eine Umstellung des Systems auf den Log-Dienst, der oft einen erheblichen Mehraufwand benötigt.

Zusammen mit dem systemd kommt das Kommando `journalctl`, mit dem Sie die Log-Dateien des systemd auslesen können. Die Informationen, die der systemd sammelt, werden immer binär gespeichert, wobei systemd nicht nur für das Logging verantwortlich ist, sondern auch das alte *init*-System der Distributionen abgelöst hat. Mehr zum Thema systemd in Bezug auf den Systemstart finden Sie in Kapitel 2, »Bootvorgang«. Zusammen mit dem systemd wird der *journal*d gestartet, der die Bootmeldung aller Dienste vom systemd entgegennimmt und auf einem laufenden System alle Meldungen von allen Diensten registriert, die über den systemd gestartet wurden. Die Informationen werden alle im Binärformat gespeichert. Der Vorteil dieses binären Formats ist der, dass Sie so in der Lage sind, alle Meldungen nach verschiedensten Kriterien filtern und weiterverarbeiten zu können. Ein weiterer Vorteil ist die sehr hohe Sicherheit vor einer Verändern des Logs auf einem System.

Den Inhalt der Dateien können Sie sich nicht mit einem Editor oder Pager anzeigen lassen. Bei den Dateien handelt es sich um gehashte Dateien. Jeder Eintrag wird mit einem eigenen Hash versehen, der immer auch aus dem vorherigen Eintrag gebildet wird. So ist eine Fälschung des Logs nahezu unmöglich. Würde jemand versuchen einen Eintrag aus der Datei zu entfernen, oder zu verändern, passen alle folgenden Hashes nicht mehr. Auch ist es möglich, die Meldungen entweder in der lokalen Zeit oder in der UTC anzeigen zu lassen. Aus diesem Grund ist es wichtig, dass Sie auf Ihren Systemen immer die richtige Zeitzone eingestellt haben. Das können Sie so wie in Listing 12.1 prüfen:

```
adminbuch:~ # timedatectl status
    Local time: So 2022-12-10 17:41:20 CET
    Universal time: So 2022-12-10 16:41:20 UTC
    RTC time: So 2022-12-10 16:41:21
    Time zone: Europe/Berlin (CET, +0100)
System clock synchronized: yes
    NTP service: inactive
    RTC in local TZ: no
root@addc-01:~# timedatectl status
    Local time: So 2022-12-10 17:41:39 CET
    Universal time: So 2021-12-10 16:41:39 UTC
    RTC time: So 2021-12-10 16:41:40
    Time zone: Europe/Berlin (CET, +0100)
System clock synchronized: yes
    NTP service: inactive
    RTC in local TZ: no
```

Listing 12.1 Prüfen der Zeitzone

Sollte die Zeitzone nicht stimmen, können Sie sie mit dem Kommando `sudo timedatectl set-timezone <zone>` setzen. Eine Liste aller Zeitzonen erhalten Sie mit dem Kommando `timedatectl list-timezones`.

12.2.1 Erste Schritte mit dem `journalctl`-Kommando

In diesem Abschnitt wollen wir Ihnen erst einmal zeigen, wie Sie sich die Meldungen auflisten lassen können. Später folgt dann noch die Verwendung von verschiedenen Filtern und Ausgabeformaten. Wenn Sie sich den Inhalt des Logs mit dem Kommando `journalctl` ohne weitere Option auflisten lassen, wird der Inhalt immer über den *Pager* ausgegeben, der standardmäßig in Ihrem System eingerichtet ist. In den meisten Fällen wird das das Kommando `less` sein. Die Einträge werden immer vom ältesten zum neuesten Eintrag angezeigt. Wollen Sie sich den Inhalt des Logs ohne die Verwendung von `less` auflisten lassen, verwenden Sie dazu die Option `--no-pager`. Dann wird das komplette Log auf einmal ausgegeben. In Tabelle 12.3 sehen Sie eine Liste der verschiedenen Ausgabemöglichkeiten.

Option	Bedeutung
<code>journalctl --no-pager</code>	komplette Ausgabe
<code>journalctl tail -n 5</code>	die letzten 5 Zeilen

Tabelle 12.3 Verschiedene Ausgabemöglichkeiten von »`journalctl`«

Option	Bedeutung
journalctl head -n 5	die ersten 5 Zeilen
journalctl --utc	UTC anstelle der lokalen Zeit
journalctl -b	alle Meldungen seit dem letzten Boot
journalctl -f	öffnet das Log und zeigt alle neuen Meldungen
journalctl -k	zeigt alle Kernelmeldungen

Tabelle 12.3 Verschiedene Ausgabemöglichkeiten von »journalctl« (Forts.)

Ob Sie sich die Meldungen der verschiedenen Systemstarts auflisten lassen können, ist abhängig davon, ob der *journald* so konfiguriert ist, dass alle Meldungen von allen Systemstarts gespeichert werden. Sie können festlegen, ob *journald* alle Meldungen speichert oder nur die nach dem aktuellen Systemstart. Alle Meldungen von älteren Systemstarts gehen dann verloren. Das Verhalten wird durch zwei Parameter bestimmt: zum einen über die Datei */etc/systemd/journal.conf* und dort durch den Parameter *Storage=* und zum anderen über das Vorhandensein des Verzeichnisses */var/log/journal*. Der Parameter *Storage* kann die folgenden Werte enthalten:

- ▶ *volatile*
Bei dieser Einstellung werden alle Log-Einträge nur im Speicher gehalten, es werden keine Log-Einträge auf der Festplatte gespeichert.
- ▶ *persistent*
Jetzt werden alle Log-Einträge auf der Festplatte gespeichert, die Daten werden im Verzeichnis */var/log/journal* abgelegt. Ist das Verzeichnis nicht vorhanden, wird es vom System angelegt.
- ▶ *auto*
Setzen Sie den Parameter auf *auto*, dann ist die Arbeitsweise abhängig davon, ob das Verzeichnis */var/log/journal* existiert. Wenn das Verzeichnis existiert, dann werden die einzelnen Journale auf der Festplatte gespeichert. Ist das Verzeichnis nicht vorhanden, wird nur ein aktuelles Log im Arbeitsspeicher geschrieben. Bei einem Neustart geht die Information verloren. Die Einstellung *auto* ist die Standardeinstellung.

Die Einstellung *auto* ist die Standardeinstellung. Bei Suse Leap und Ubuntu ist das Verzeichnis */var/log/journal* vorhanden, und es werden somit alle Logs permanent auf die Festplatte geschrieben. Bei allen anderen Distributionen fehlt das Verzeichnis. Somit können Sie dort nur die Meldungen seit dem aktuellen Systemstart abrufen. Wollen Sie das ändern, brauchen Sie nur das Verzeichnis */var/log/journal* anzulegen.

Wenn Sie wissen wollen, welche Journale auf Ihrem System vorhanden sind, können Sie das so wie in Listing 12.2 in Erfahrung bringen:

```
root@ldap01:~# journalctl --list-boots
-1 b44fd9e0ecca49429ea7c9f6b5893e67 Sun 2022-12-11 11:02:42 \
   CET--Mon 2022-12-11 11:58:53 CET
 0 b8cc607c511945da935d7e1535937123 Sun 2022-12-11 12:01:24 \
   CET--Mon 2022-12-11 12:01:41 CET
```

Listing 12.2 Auflisten aller Logs

Wenn Sie sich jetzt das Protokoll des Logs -1 ansehen wollen, können Sie das mit dem Kommando `journalctl -b -1` erreichen.

12.2.2 Filtern nach Zeit

Bis jetzt haben Sie nur gesehen, wie Sie sich das gesamte Log oder einzelne Logs ansehen können. Aber eine der wichtigsten Filtermöglichkeiten ist das Filtern nach einer bestimmten Zeit oder einem bestimmten Zeitraum. Dazu sehen Sie in Listing 12.3 einige Beispiele:

```
root@ldap01:~# journalctl --since "2022-12-11 12:04:00"
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:04:07 CET. --
```

```
Dec 11 12:04:07 ldap01 systemd[1]: Stopping LSB: OpenLDAP standalone \
server (Lightweight Directory Access Protocol)...
```

```
[...]
```

```
Dec 11 12:04:07 ldap01 slapd[554]: Starting OpenLDAP: slapd.
```

```
[...]
```

```
root@ldap01:~# journalctl --until "2022-012-11 12:04:00"
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:06:52 CET. --
Dec 11 11:02:42 ldap01 kernel: ...
```

```
root@ldap01:~# journalctl --since "2022-12-11 12:00:00" --until "2022-12-11 12:04:00"
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Mon 2021-01-11 12:06:52 CET. --
Jan 11 12:01:24 ldap01 kernel: ...
```

Listing 12.3 Filtern nach Zeit

Wie Sie an den drei Beispielen sehen, können Sie sowohl eine Anfangszeit für die Ausgabe des Logs wählen als auch eine Endzeit des Logs. Auch eine Kombination aus beiden ist möglich, wie Sie im dritten Beispiel in Listing 12.3 sehen.

Aber das sind noch nicht alle Möglichkeiten, die Sie haben. Auch intuitive Zeitangaben sind möglich, wie Sie den Beispielen aus Listing 12.4 entnehmen können:


```

root@ldap01:~ # journalctl --since yesterday
root@ldap01:~ # journalctl --since 09:00 --until "1 hour ago"
root@ldap01:~ # journalctl --since "2021-12-11 17:15:00" --until now

```

Listing 12.4 Weitere Beispiele für die Zeit

So lassen sich Fehler genau eingrenzen. Weitere Schlüsselwörter, die Sie verwenden können, sind `today` und `tomorrow`. Bei den Zeiten können Sie auch noch mit `+` und `-` arbeiten, wie Sie in Listing 12.5 sehen können:

```

root@ldap01:~ # journalctl --since "2016-07-04 17:15:00" --until "+1 hour"
root@ldap01:~ # journalctl --since "2016-07-04 17:15:00" --until "-1 minute"

```

Listing 12.5 Beispiele mit »+« und »-«

12.2.3 Filtern nach Diensten

Natürlich lässt sich das Log auch nach Meldungen von Diensten durchsuchen, die vom `systemd` verwaltet werden. Wenn Sie auf einem `openLDAP`-Server alle Meldungen des `slapd` ausfiltern wollen, dann können Sie die Meldungen so wie in Listing 12.6 ausfiltern:

```

root@ldap01:~# journalctl -u slapd.service
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:11:17 CET. --
Jan 11 11:02:45 ldap01 systemd[1]: Starting LSB: OpenLDAP ...
Jan 11 11:02:46 ldap01 slapd[504]: @(#)...
Jan 11 11:02:46 ldap01 slapd[506]: slapd starting
Jan 11 11:02:46 ldap01 slapd[496]: Starting OpenLDAP: slapd.
[...]

```

Listing 12.6 Filtern von Servicemeldungen

Gerade wenn der Server schon längere Zeit läuft, kann es sinnvoll sein, den Zeitraum einzugrenzen. Sie die Filterung des Dienstes auch mit den Zeitfiltern aus dem vorherigen Abschnitt koppeln. Listing 12.7 zeigt dazu ein Beispiel:

```

root@ldap01:~# journalctl -u slapd.service --since \
                "2021-01-11 12:04:00" --until "-1 minute"
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:17:01 CET. --

```

Listing 12.7 Koppelung von Service- und Zeitfiltern

Auch ein Auflisten aller Einträge, die zu einer bestimmten `PID` gehören, ist möglich, wie Listing 12.8 zeigt:

```

root@ldap01:~# ps ax | grep slapd | awk '{print $1}'
560

```

```
root@ldap01:~# journalctl _PID=560
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:20:25 CET. --
Jan 11 12:04:07 ldap01 slapd[560]: slapd starting
```

Listing 12.8 Log-Einträge nach PID

Auch das Auflisten aller Logs zu einem bestimmten Benutzer oder einer Gruppe ist möglich, so wie Sie es in Listing 12.9 sehen:

```
root@ldap01:~# id openldap
uid=106(openldap) gid=112(openldap) Gruppen=112(openldap)
```

```
root@ldap01:~# journalctl _UID=106 --since today
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:21:36 CET. --
Jan 11 11:02:46 ldap01 slapd[506]: slapd starting
```

Listing 12.9 Auflisten aller Einträge zu einem bestimmten Benutzer oder einer Gruppe

Bei den letzten Beispielen haben wir die *Metadatenfelder* verwendet. Diese Metadatenfelder werden vom Journal dazu verwendet, bestimmte Meldungen zu indizieren, schon während sie in das Log eingetragen werden. So können Sie an der Stelle schon gezielt nach bestimmten Parametern suchen. Alle verwendeten Metadatenfelder können Sie sich mit dem Kommando `journalctl -o verbose -n` anzeigen lassen.

12.2.4 Kernelmeldungen

Zum Abschluss möchten wir hier noch einmal etwas genauer auf die Kernelmeldungen eingehen, da diese Meldungen auch noch auf eine interessante Art gefiltert werden können. Natürlich können Sie die folgenden Filter auch wieder mit allen vorher besprochenen Möglichkeiten kombinieren.

Wie schon in Tabelle 12.3 beschrieben, können Sie sich alle Kernelmeldungen mit dem Kommando `journalctl -k` anzeigen lassen. Damit werden Ihnen immer alle Meldungen seit dem letzten Bootvorgang angezeigt. Wollen Sie aber die Meldungen eines vorherigen Bootvorgangs sehen, können Sie dafür so vorgehen wie in Listing 12.10:

```
root@ldap01:~# journalctl -k -b -1
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:28:18 CET. --
Jan 11 11:02:42 ldap01 kernel: Linux version ...
Jan 11 11:02:42 ldap01 kernel: Command line: ...
```

Listing 12.10 Kernelmeldungen eines bestimmten Bootvorgangs

Auch eine Filterung auf bestimmte Level der Meldungen ist möglich. Ein Beispiel dazu sehen Sie in Listing 12.11:

```
root@ldap01:~# journalctl -p err -b
-- Logs begin at Sun 2022-12-11 11:02:42 CET, end at Sun 2022-12-11 12:35:25 CET. --
Jan 11 12:01:24 ldap01 systemd-fstab-generator[207]: Failed ...
```

Listing 12.11 Meldungen eines bestimmten Levels filtern

Die Option `-b` zeigt nur die Meldungen nach dem letzten Bootvorgang an. Die Meldungen können Sie nach den in Tabelle 12.2 aufgelisteten Leveln filtern.

12.2.5 Einrichten eines Log-Hosts

Wie auch mit einigen der älteren Log-Daemons lässt sich mit dem *journal*d ein *Log-Host* einrichten, um alle Meldungen Ihrer Server zentral an einer Stelle zu speichern. Die Übertragung der Daten wird dabei über TLS verschlüsselt, sodass ein Mitlesen mit einem Netzwerksniffer nicht möglich ist. Für das Beispiel werden wir einen LDAP-Server als Log-Client einrichten, der seine Log-Informationen dann auf den Log-Host speichert. Auf allen Clients und dem Server installieren Sie das Paket *systemd-journal-remote*.

Achten Sie bei der Einrichtung des Servers darauf, dass ausreichend Festplattenplatz vorhanden ist. Je mehr Clients die Logs dort ablegen und je länger Sie die Logs speichern wollen, umso mehr Plattenplatz wird benötigt. Damit der Server, für den Fall, dass der Plattenplatz nicht ausreicht, nicht abstürzt, lagern Sie das Verzeichnis `/var` auf eine eigene Partition aus. Die Standardgröße einer Log-Datei ist 8 MB, auch wenn noch keine Meldungen im Log eingetragen sind.

Im ersten Schritt aktivieren Sie den Dienst und den dazugehörigen Socket auf dem Server, über den sich die Clients verbinden. Listing 12.12 zeigt die entsprechenden Kommandos:

```
root@loghost:~# systemctl enable --now systemd-journal-remote.socket
Created symlink /etc/systemd/system/sockets.target.wants/systemd-
journal-remote.socket -> /lib/systemd/system/systemd-journal-remote.socket.
root@loghost:~# systemctl enable systemd-journal-remote.service
```

Listing 12.12 Starten der Dienste auf dem Server

Im ersten Kommando wird der Dienst durch die Option `--now` sofort gestartet. Der zweite Dienst wird erst gestartet, wenn Sie das TLS-Zertifikat bereitgestellt haben. Der Port, der hier verwendet wird, ist der Port 19532. Prüfen Sie mit dem Kommando `ss`, ob der Port geöffnet wurde.

Erstellen Sie jetzt, ein Zertifikat für den Server falls noch keines vorhanden ist.

Auf allen Clients die sich mit dem Log-Host verbinden sollen, aktivieren Sie jetzt auch den *systemd-journal-remote.service*. Auch auf den Clients wird der Dienst aber noch nicht gestartet.



Einrichten des Servers

Die Konfiguration des Log-Servers finden Sie in der Datei `/etc/systemd/journal-remote.conf`. In der Datei sind schon bestimmte Vorgaben vorhanden, die aber alle auskommentiert sind. In Listing 12.13 sehen Sie die Datei für unser Beispiel:

```
[Remote]
# Seal=false
SplitMode=host
ServerKeyFile=/etc/ssl/zertifikate/loghost-key.pem
ServerCertificateFile=/etc/ssl/zertifikate/loghost-cert.pem
TrustedCertificateFile=/etc/ssl/zertifikate/demoCA/cacert.pem
```

Listing 12.13 Konfigurationsdatei des Servers

Die Variablen haben die folgenden Bedeutungen:

- ▶ **Seal**
Wenn Sie maximale Sicherheit haben wollen, was das Manipulieren der Einträge angeht, können Sie diese Variable aktivieren und auf `true` setzen. Dann werden die einzelnen Einträge zusätzlich signiert.
- ▶ **SplitMode**
Alle Einträge werden aufgrund des Hostnamens in eigene Dateien gespeichert. Das verbessert die Übersichtlichkeit über die Daten, gerade wenn Sie viele Hosts auf dem Server loggen wollen.
- ▶ **ServerKeyFile**
Der Pfad und der Name des Keyfiles Ihres Zertifikats.
- ▶ **ServerCertificateFile**
Der Name und der Pfad zum Zertifikat.
- ▶ **TrustedCertificateFile**
Der Name und Pfad zum root-Zertifikat. Ohne diese Angabe kann die Gültigkeit der Zertifikate nicht geprüft werden.

Damit die Zertifikate auch vom log-Daemon genutzt werden können, ist es notwendig, dass Sie die Rechte wie in Listing 12.14 anpassen:

```
root@loghost:~# chgrp systemd-journal-remote /etc/ssl/zertifikate/loghost-*
root@loghost:~# chmod 640 /etc/ssl/zertifikate/loghost-*
```

Listing 12.14 Die Rechte für die Zertifikate auf dem Server anpassen



Sollten Sie das Zertifikat auch noch für andere Dienste benötigen, setzen Sie die erweiterten ACLs für die Gruppe. Erst jetzt können Sie den Dienst mit dem Kommando `systemctl start systemd-journal-remote.service` starten. Danach ist der Log-Host bereit, die Log-Daten der Clients anzunehmen.

Einrichten des Clients

Nachdem der Server im vorherigen Abschnitt eingerichtet wurde, geht es jetzt darum, den ersten Client anzubinden, der seine Log-Daten an den Log-Server schickt.

Für die Übermittlung der Daten wird ein eigener Benutzer benötigt. Legen Sie einen neuen Benutzer an, indem Sie das Kommando aus Listing 12.15 übernehmen:

```
root@loghost:~# adduser --system --home /run/systemd --no-create-home \
    --disabled-login --group systemd-journal-upload
Lege Systembenutzer »systemd-journal-upload« (UID 107) an ...
Lege neue Gruppe »systemd-journal-upload« (GID 113) an ...
Lege neuen Benutzer »systemd-journal-upload« (UID 107) mit \
    Gruppe »systemd-journal-upload« an ...
Erstelle Home-Verzeichnis »/run/systemd« nicht.
```

Listing 12.15 Anlegen eines Benutzers für den Log-Dienst

12

Bei dem neuen Benutzer handelt es sich um einen Systembenutzer, der sich nicht lokal anmelden kann. Vergeben Sie auch kein Passwort für diesen Benutzer. Sorgen Sie auf dem Client dafür, dass das Zertifikat von der Gruppe *systemd-journal-remote* gelesen werden kann.

Auch auf dem Client folgt jetzt die Anpassung der Konfiguration, dieses Mal in der Datei */etc/systemd/journal-upload.conf*. Auch auf dem Client ist die Datei schon vorhanden, aber alle Optionen sind noch auskommentiert. In Listing 12.16 sehen Sie die an unser Beispiele angepasste Datei:

```
[Upload]
URL=https://loghost.example.net:19532
ServerKeyFile=/etc/ssl/zertifikate/ldap01-key.pem
ServerCertificateFile=/etc/ssl/zertifikate/ldap01-cert.pem
TrustedCertificateFile=/etc/ssl/zertifikate/demoCA/cacert.pem
```

Listing 12.16 Anpassung der Client-Konfiguration

Im Unterschied zur Konfiguration des Servers wird hier die URL angegeben, unter der der Server erreichbar ist. Die drei Zeilen für die Zertifikate sind identisch mit denen auf dem Server. Starten Sie den Log-Daemon jetzt mit dem Kommando `systemctl restart systemd-journal-upload.service` auf dem Client.

Testen der Log-Umgebung

Wechseln Sie auf den Log-Server, und lassen Sie sich den Inhalt des Verzeichnisses */var/log/journal/remote* anzeigen. Dort sehen Sie jetzt eine neue Datei. Der Name der Datei enthält den vollständigen Namens des Clients, wie er im Zertifikat genannt wird. Einen Eintrag zu unserem Beispiele sehen Sie in Listing 12.17:

```
root@loghost:~# ls /var/log/journal/remote/
'remote-C=DE,ST=SH,L=St.\x20Michel,O=Adminbuch,OU=CA,CN=ldap01.example.net.journal'
```

Listing 12.17 Anzeigen des Dateinamens auf dem Log-Server



Leerzeichen zusätzlich entwerfen

Sollten Sie, so wie im Beispiel, ein Leerzeichen im Namen des Zertifikats haben, dann wird das Leerzeichen durch die Zeichenfolge `\x20` ersetzt. Der Backslash muss später beim Einlesen der Datei zusätzlich mit einem Backslash entwertet werden, auch wenn Sie den Pfad in gerade Hochkommata setzen.

Ist die Datei vorhanden, hat alles geklappt und der Server nimmt Meldungen vom Client entgegen. Sie können jetzt warten, bis der Client eine Nachricht generiert und auf den Server überträgt. Sie können aber auch selber eine Testmeldung mit dem Kommando `logger` generieren, so wie in Listing 12.18:

```
root@ldap01:~# logger -p syslog.debug "Das ist ein Test vom Client ldap01"
```

Listing 12.18 Erstellen einer Log-Meldung auf dem Client

Auf dem Log-Server können Sie jetzt das Log genau so auswerten, wie wir es in den Abschnitten vorher beschrieben haben, aber mit einem Unterschied: Als Quelle geben Sie die Datei des Clients an, den Sie auswerten wollen. In Listing 12.19 sehen Sie das Kommando und das Ergebnis der durch `logger` erzeugten Meldung:

```
root@loghost:~# journalctl --file='/var/log/journal/remote/remote-C=DE,ST=SH,\
    L=St.\x20Michel,O=Adminbuch,OU=CA,CN=ldap01.example.net.journal'
[...]
Jan 11 19:48:18 ldap01 root[1081]: Das ist ein Test vom Client ldap01
```

Listing 12.19 Auswertung des Client-Logs

12.3 Der Klassiker: Syslogd



Außer CentOS stellen alle anderen hier verwendeten Distributionen den `syslogd` noch zur Verfügung. Die Konfigurationsdatei `syslog.conf` steuert das Verhalten des Syslog-Servers. Sie findet sich unter `/etc` oder `/etc/syslog`, wenn Sie den `syslogd` zusätzlich installiert haben. In Listing 12.20 folgt ein Ausschnitt mit typischen Einträgen:

```
kern.*      -/var/log/kern.log
mail.*     -/var/log/mail.log
user.*     -/var/log/user.log
```

Listing 12.20 Ausschnitt aus der »`syslog.conf`«

Auf der linken Seite steht der Selektor. *Facility* und *Severity* sind durch einen Punkt getrennt. Wenn für den beschriebenen Selektor eine Nachricht eintrifft, wird die auf der rechten Seite aufgeführte *Action* ausgelöst. In den Beispielen aus Listing 12.20 werden die Nachrichten in die angegebenen Dateien geschrieben. Das Minuszeichen vor dem Dateipfad bedeutet, dass die Nachrichten nicht sofort beim Eintreffen in die Datei wandern, sondern zunächst gepuffert werden – bei einem hohen Nachrichtenaufkommen verhindert die Pufferung ungebührliche Hektik auf den Festplatten.

Um genau die Meldungen im Log zu finden, die von Bedeutung sind, lassen sich *Facility* und *Severity* auf mehrere Arten filtern oder zusammenfassen:

- ▶ `kern.warn -/var/log/kern.log`
Kernmeldungen mit Log-Level `warn` oder höher werden geschrieben.
- ▶ `kern.=warn -/var/log/warnings.log`
Kernmeldungen mit genau dem Log-Level `warn` werden geschrieben.
- ▶ `news,kern.warn -/var/log/warnings.log`
Für die *Facilitys* `news` und `kern` werden alle Meldungen mit dem Log-Level `warn` oder höher geschrieben.
- ▶ `mail.warn;kern.=crit -/var/log/kaputt.log`
Für die *Facility* `mail` werden alle Nachrichten mit dem Log-Level `warn` oder höher geloggt, zusätzlich alle Meldungen der Kernel-*Facility*, die genau den Log-Level `crit` haben.
- ▶ `*.warn;news.none -/var/log/nonews.log`
Alle Nachrichten aller *Facilitys* außer `news` mit dem Log-Level `warn` oder höher werden geloggt.
- ▶ `mail.=!debug -/var/log/mail.nodebug.log`
Erfasst werden alle Nachrichten der *Facility* `mail` mit Ausnahme der Debug-Meldungen.

Das Schreiben in eine Datei ist natürlich nicht die einzige *Action*, die ausgelöst werden kann. Weitere Aktionen finden Sie in Tabelle 12.4.

Action	Bedeutung
/pfad/dateiname	Eintrag in Datei
-/pfad/dateiname	Eintrag in Datei mit Pufferung
@hostname	Nachricht an Host weiterleiten
pipe	in die Pipe schreiben

Tabelle 12.4 Aktionen

Action	Bedeutung
username	auf das Terminal des Users ausgeben
*	»Wall«: auf alle Terminals ausgeben

Tabelle 12.4 Aktionen (Forts.)

12.4 Syslog-ng

Syslog-ng benutzt eine gänzlich andere Konfigurationssyntax als Syslogd. Sie erinnert an C- und Java-Code und wirkt durch die vielen Klammerungen auf den ersten Blick verwirrend, ist aber tatsächlich nicht schwer zu verstehen. Ein großer Vorteil des Syslog-ng ist, dass Sie die Darstellung der Meldungen im Logfile selbst definieren können. Da die Dienste die Darstellung der Zeilen oft anders vornehmen, zum Beispiel in welchem Format Datum und Uhrzeit angezeigt werden, ist es oft schwer, Log-Meldungen aus einem Logfile zu filtern. Wenn Sie aber alle Darstellungen selbst definieren, ist das Filtern erheblich einfacher. Auch hier der Hinweis, dass CentOS den Syslog-ng nicht mehr bereitstellt. Die Konfiguration finden Sie per Default in der Datei `/etc/syslog-ng/syslog-ng.conf`. Dort kann es sechs verschiedene Arten von Einträgen geben:

- ▶ `options`: In diesem Element werden globale Optionen festgelegt.
- ▶ `source`: Definiert die Quellen, aus denen Syslog-ng die Log-Meldungen bezieht.
- ▶ `destination`: Hier werden die Zielobjekte benannt, die die Nachrichten aufnehmen sollen. Im Regelfall sind dies einfache Textdateien.
- ▶ `filter`: Auf eingehende Meldungen können Filterregeln angewandt werden, um erwünschte von unerwünschten Nachrichten zu trennen.
- ▶ `log`: Im Log-Element werden Quellen, Ziele und Filter kombiniert, und das eigentliche Logging wird aktiviert. Während die zuvor beschriebenen Objekte nur Definitionen sind, löst das Log-Element – und nur dieses – auch eine Aktion aus.
- ▶ `template`: Im `template` können Sie die Nachricht formatieren, um eine einheitliche Ausgabe aller Meldungen zu erhalten.

12.4.1 Der »options«-Abschnitt

Die globalen Optionen sind distributionsseitig fast immer auf sinnvolle Werte gesetzt. Trotzdem lohnt es sich, gerade beim Aufbau einer größeren Logging-Umgebung, einigen Punkten etwas mehr Aufmerksamkeit zu schenken. Ein typischer `options`-Abschnitt kann folgendermaßen aussehen:


```
[...]
options{
    owner(syslogng);
    group(syslogng);
    perm(0640);
    create_dirs(yes);
    dir_perm(0755);
    sync(100);
    log_fifo_size(2048);
    use_dns(yes);
    dns_cache_size(512);
    dns_cache_expire(3600);
    dns_cache_expire_failed(1200);
    log_msg_size(4096);
}
```

Listing 12.21 »options«-Abschnitt der »syslog-ng.conf«

Die einzelnen Einträge haben die folgende Bedeutung:

- ▶ `owner(userid)` und `group(groupid)` definieren den Benutzernamen und die Gruppenzugehörigkeit.
- ▶ `perm()` setzt die Dateirechte für alle Log-Dateien, die Syslog-ng schreibt.
- ▶ `create_dirs(yes)` erlaubt dem Syslog-Daemon, fehlende Verzeichnisse anzulegen. Deren Benutzer- und Gruppeneinstellungen sowie die Dateisystemrechte können Sie mit den Direktiven `dir_owner()`, `dir_group()` und `dir_perm()` festlegen.
- ▶ `sync(100)`; bedeutet, dass Syslog-ng seine Meldungen erst auf die Platte schreibt, wenn 100 Nachrichten beisammen sind. Auf stark belasteten Systemen wie dedizierten Log-Servern kann eine solche Einstellung viel I/O-Last sparen. Der Standardwert hingegen ist 0, es wird also unmittelbar geschrieben. Sicherheitsfanatiker sorgen lieber für ein leistungsfähiges Plattensubsystem und lassen den Wert immer auf 0, denn im größten anzunehmenden Murphy-Fall brennt das System ab, während exakt 99 Nachrichten im Puffer sind – darunter auch die Meldung, die den Absturz erklärt.

Um bei einer plötzlich hereinbrechenden Meldungsflut keine Daten zu verlieren, benötigt Ihr System dennoch einen gewissen Notfallpuffer. Schließlich benutzt Syslog serienmäßig UDP, verlorene Daten können also nicht neu angefordert werden. Diesen Notnagel definieren Sie mit `log_fifo_size(2048)`; Die Zahl 2048 gibt die Anzahl der Zeilen an, die maximal in die Ausgabewarteschlange passen.

- ▶ `use_dns(yes)` erlaubt die Namensauflösung unbekannter Clients. Falls die Namensauflösung genutzt wird, ist es sinnvoll, die Ergebnisse in einem Cache zwischenspeichern. Der Cache kann mit `dns_cache(yes)` aktiviert werden.



- ▶ Der Eintrag `dns_cache_size(512)` legt fest, dass nicht mehr als 512 Namen hineinpassen. Wie lange die Ergebnisse im Cache bleiben, wird mit `dns_cache_expire(sek)` und `dns_cache_expire_failed(sek)` gesteuert. `sek` gibt die Zeit in Sekunden bis zur Löschung aus dem Cache an.
- ▶ `log_msg_size(4096)` erlaubt es, Nachrichten zu verarbeiten, die die im Syslog-Protokoll vorgegebene Länge von maximal 1.024 Zeichen überschreiten.

12.4.2 Das »source«-Objekt

In diesem Abschnitt definieren Sie, aus welchen Quellen Syslog-ng Nachrichten empfangen soll. Er sieht auf den meisten Linux-Systemen so aus wie in Listing 12.22:

```
source s_quellen{
    internal();
    file("/proc/kmsg");
    unix-stream("/dev/log");
}
```

Listing 12.22 Source-Definitionen in der »syslog-ng.conf«

Zunächst wird ein Source-Objekt mit dem generischen Namen `s_quellen` angelegt. Darin sind drei Quellen angegeben:

- ▶ `internal()`
ist die Quelle für Nachrichten, die Syslog-ng selbst erzeugt.
- ▶ `file("/proc/kmsg")`
liefert Log-Meldungen des Kernels.
- ▶ `unix-stream("/dev/log")`
dient als Quelle für alle anderen lokal generierten Nachrichten.

Natürlich kann Syslog-ng auch Nachrichten über das Netzwerk empfangen. Dieser Funktionalität ist im weiteren Verlauf ein eigener Abschnitt gewidmet.

12.4.3 Das »destination«-Objekt

Das Zielobjekt für Log-Nachrichten ist in der Regel eine Datei. Im folgenden Beispiel bekommt das Objekt den Namen `d_systemlogs`:

```
destination d_systemlogs{
    file("/var/log/systemlogs");
}
```

Listing 12.23 Ein Zielobjekt in der »syslog-ng.conf«

Innerhalb der Objektdefinition können Eigenschaften des Objekts festgelegt werden, die von den Einstellungen im `options`-Abschnitt abweichen. Statt mit einer `0640`-Berechtigung können Sie die Datei auch mit einer `0755`-Berechtigung anlegen lassen:

```
destination d_systemlogs{
    file("/var/log/systemlogs"
        perm(0755));
}
```

Listing 12.24 Zielobjekt mit abweichender Dateiberechtigung

Auch das Format, in dem die Meldungen in die Log-Datei geschrieben werden, ist nicht in Stein gemeißelt. Es lässt sich mit der `template`-Direktive anpassen:

```
destination d_systemlogs{
    file("/var/log/systemlogs"
        perm(0755)
        template("$DATE $FULLHOST [$FACILITY.$LEVEL] $MESSAGE\n") );
}
```

Listing 12.25 Zielobjekt mit benutzerdefiniertem Format

Tabelle 12.5 zeigt die dabei zulässigen Variablen.

Variable	Bedeutung
DAY	Tag des Monats, zweistellig (zum Beispiel 08)
MONTH	Monat, zweistellig
YEAR	Jahreszahl, vierstellig
WEEKDAY	Wochentag, englisch, dreistellig (zum Beispiel Tue)
HOURL	Stunde im 24-Stunden-Format, zweistellig
MIN	Minuten, zweistellig
SEC	Sekunden, zweistellig
FACILITY	die Log-Facility
LEVEL (auch PRIORITY)	der Log-Level
HOST	Hostname ohne Domain
FULLHOST	vollständiger Hostname, zum Beispiel <i>host.domain.tld</i>

Tabelle 12.5 Variablen zur Anpassung des Ausgabeformats

Variable	Bedeutung
PROGRAM	Name des Programms, das die Meldung geschickt hat
MSG (auch MESSAGE)	die eigentliche Meldung

Tabelle 12.5 Variablen zur Anpassung des Ausgabeformats (Forts.)

Die gleichen Variablen funktionieren auch in der `file`-Direktive. Damit ist es beispielsweise möglich, für jeden Host, von dem Log-Daten empfangen werden, ein eigenes Verzeichnis anzulegen:

```
destination d_systemlogs{
    file("/var/log/$HOST/systemlogs"
        perm(0755)
        template("$DATE $FULLHOST [$FACILITY.$LEVEL] $MESSAGE\n") );
}
```

Listing 12.26 Ein Verzeichnis pro Host

Mit dem Schlüsselwort `usertty` können Meldungen statt in eine Datei auf das Terminal eines Benutzers ausgegeben werden:

```
destination d_charlysterminal{
    usertty(charly);
}
```

Listing 12.27 Auf ein Terminal ausgeben

Wird anstelle des Usernamens ein Asterisk (*) gesetzt, geht die Meldung an alle Terminals:

```
destination d_allterminals{
    usertty(*);
}
```

Listing 12.28 Auf alle Terminals ausgeben

12.4.4 Das »filter«-Objekt

Wie Sie bereits gesehen haben, bietet der klassische Syslog-Daemon die Möglichkeit, nach *Facility* und *Log-Level* zu sortieren. Syslog-ng verfeinert diese Möglichkeiten und benutzt dazu das *filter*-Objekt. In einem ersten Beispiel wollen wir aus dem eingehenden Meldungsstrom alle Nachrichten herausholen, die von der Mail-Facility kommen. Geben Sie diesem Objekt zum Beispiel den Namen `f_allmail`. Prinzipiell sind Sie bei der Vergabe der Objektnamen frei, es hat sich jedoch eingebürgert, die Namen von Filterobjekten mit `f_` beginnen zu

lassen. Einige Distributionen lassen in der Standardkonfiguration analog dazu die Namen von Source- und Destination-Objekten mit `s_` und `d_` beginnen.

```
filter f_allmail { facility (mail); };
```

Im nächsten Schritt schränken Sie diesen Filter so ein, dass er nur noch die Log-Level `err` und `crit` erfasst:

```
filter f_brokenmail { facility (mail) and level (err, crit); };
```

Neben `and` können auch `or` und `not` verwendet werden, um den Filterausdruck weiter zu verfeinern. Der folgende Filter erfasst alle Log-Level von `err` bis `emerg` für alle Facilitys mit Ausnahme von `news` und `cron`:

```
filter f_nonewscron { level (err..emerg) and not facility (news, cron); };
```

Neben `facility` und `level` gibt es noch weitere Elemente, die als Filterkriterium dienen können. Schauen wir sie uns der Reihe nach an:

► `host (regex)`

Wenn der reguläre Ausdruck `regex` auf den Namen des versendenden Rechners passt, wird diese Meldung von der Filterregel erfasst. Wollen Sie beispielsweise alle Meldungen haben, die vom Host »webserver2« kommen, definieren Sie das folgende Objekt:

```
filter f_ws2 { host ("^webserver2$"); };
```

In den meisten Fällen wird auch die einfachere Schreibweise

```
filter f_ws2 { host ("webserver2"); };
```

ausreichen. Das erste Beispiel passt exakt auf den Ausdruck `webserver2`, das zweite erfasst auch Ausdrücke, in denen diese Zeichenfolge vorkommt – es würde also auch auf den Hostnamen »dmz3-webserver2-kunde« zutreffen.

► `match (regex)`

Hier werden alle Nachrichten erfasst, bei denen der reguläre Ausdruck `regex` im Meldungstext vorkommt. Wollen Sie alle Meldungen haben, in denen das Wort »failed« vorkommt, definieren Sie folgenden Filter:

```
filter f_failed { match (failed); };
```

► `program (regex)`

Hier wird auf den Programmnamen gefiltert. Der Ausdruck erfasst alle Nachrichten, die von einem Programm kommen, auf dessen Namen der reguläre Ausdruck `regex` zutrifft.

► `filter (objektname)`

Mit diesem Ausdruck können bereits definierte Filterobjekte in ein neues Filterobjekt eingebunden werden. Möchten Sie zum Beispiel alle Meldungen aus der Facility `mail` sehen, in denen das Wort »failed« vorkommt, sieht die Regel so aus:

```
filter f_mailfailed { facility (mail) and filter(f_failed); };
```

Die Verkettung mehrerer komplexer Filterobjekte ist möglich, aber unübersichtlich und führt relativ schnell zu Datenspaghetti in den Logfiles. Nutzen Sie lieber mehrere einfache Konstrukte als wenige komplexe.

12.4.5 Das »log«-Objekt

Bisher haben wir nur Objekte für Nachrichtenquellen, -filter und -ziele definiert, ohne wirklich ein »echtes« Logging zu bekommen. Das ändert sich mit dem log-Objekt. Mit ihm verknüpfen Sie Quelle, Ziel und Filter zu einer Logging-Anweisung:

```
log {
    source(s_allmessages);
    filter(f_failed);
    destination (d_failedlog);
}
```

Listing 12.29 Ein »log«-Objekt

Erst wenn ein gültiges log-Objekt existiert, werden die Nachrichten auch tatsächlich in eine Datei geschrieben.

12.5 Rsyslog

Rsyslog ist neben Syslog-ng eine weitere Alternative zum klassischen Syslogd. Im Übrigen ist seine Konfigurationssyntax mit der von Syslogd identisch. Rsyslog-Funktionen, die über diesen Umfang hinausgehen, werden einfach hinzudefiniert. Um die klassische von der erweiterten Konfiguration zu trennen, werden oft zwei oder mehr Konfigurationsdateien angelegt. Eine dieser Dateien, meist */etc/rsyslog.conf*, enthält den grundlegenden und Syslogd-kompatiblen Teil sowie ein Include-Statement, mit dem weitere Dateien eingelesen werden (zum Beispiel: `$IncludeConfig /etc/rsyslog.d/*.conf`). Bei der erweiterten Konfigurationssyntax sind, wie bei Syslog-ng, die Filtermöglichkeiten von besonderem Interesse. Neben den einfachen Facility- bzw. Log-Level-Filtern, die die klassische Syslog-Syntax bietet, offeriert Rsyslog eigenschafts- und ausdrucksbasierte Filter.

12.5.1 Eigenschaftsbasierte Filter

Ein eigenschaftsbasierter Filter hat die folgende Form:

```
:eigenschaft, vergleichsoperation, "wert"
```

Ein Beispiel aus der Praxis sieht so aus:

```
:programmname, contains, "ipop3d/var/log/mail/ipop3d.log"
```

Diese Zeile durchforstet den Meldungsstrom nach Programmnamen – das ist die »Eigenschaft«, von der diese Filtermethode ihren Namen hat – und sorgt für eine Reaktion, wenn der Programmname die Zeichenkette `ipop3d` enthält (`contains`). Die Meldung wird dann in die Datei `/var/log/mail/ipop3d.log` geschrieben. Weitere Eigenschaften, in denen nach dem gleichen Muster gefiltert werden kann, können Sie Tabelle 12.6 entnehmen.

Variable	Bedeutung
<code>pri-text</code>	Facility und Log-Level, durch Punkt getrennt (zum Beispiel <code>mail.err</code>)
<code>syslogfacility-text</code>	nur die Facility
<code>syslogseverity-text</code>	nur die Severity (der Log-Level)
<code>programname</code>	der Name des Programms, das die Nachricht schickt
<code>hostname</code>	Hostname des Servers, auf dem die Meldung generiert wurde
<code>msg</code>	die eigentliche Nachricht
<code>timereported</code>	Zeitstempel der Nachricht

Tabelle 12.6 Die wichtigsten Eigenschaften, die sich zum Filtern eignen

Natürlich ist auch `contains` nicht die einzig mögliche Vergleichsoperation. Insgesamt stehen Ihnen fünf Vergleichsoperationen zur Verfügung, die in Tabelle 12.7 aufgelistet sind.

Variable	Bedeutung
<code>contains</code>	prüft, ob die Zeichenkette im Eigenschaftsfeld enthalten ist
<code>isequal</code>	prüft, ob die Zeichenkette exakt identisch ist
<code>startswith</code>	prüft, ob das Eigenschaftsfeld mit der angegebenen Zeichenkette beginnt
<code>regex</code>	vergleicht das Eigenschaftsfeld mit einem regulären Ausdruck nach POSIX BRE
<code>ereregex</code>	vergleicht das Eigenschaftsfeld mit einem regulären Ausdruck nach POSIX ERE

Tabelle 12.7 Vergleichsoperationen

12.5.2 Ausdrucksbasierte Filter

Ausdrucksbasierte Filter ermöglichen unter anderem Fallunterscheidungen mit `if...then` sowie Verknüpfungen mit `and`, `or` und `not`.

Als Beispiel sollen alle Mails, die

- ▶ der Facility »mail« angehören
- ▶ und den String »device« enthalten
- ▶ und außerdem entweder den String »error« oder »fatal« enthalten,
- ▶ aber nicht mit dem String »NOQUEUE« beginnen,

in die Datei `/var/log/maildevice-err.log` geschrieben werden:

```
if $syslogfacility-text == 'mail' and $msg contains 'device' and ($msg contains  
'error' or $msg contains 'fatal') and not ($msg startswith 'NOQUEUE')  
then /var/log/maildevice-err.log
```

Listing 12.30 Ausdrucksbasierter Filter

Leider müsste all dies eigentlich auf einer einzigen Zeile stehen. In der Konfigurationsdatei kann man sich jedoch der besseren Lesbarkeit halber mit dem Backslash behelfen. Diese Konfiguration wird als eine einzige Zeile interpretiert:

```
if $syslogfacility-text == 'mail' \  
    and $msg contains 'device' \  
    and ($msg contains 'error' or $msg contains 'fatal') \  
    and not ($msg startswith 'NOQUEUE') \  
then /var/log/maildevice-err.log
```

Listing 12.31 Bessere Lesbarkeit durch Backslashes

Glücklicherweise ist die Komplexität der Filterregeln dadurch begrenzt, dass Rsyslog keine Konfigurationszeilen mit mehr als 4.000 Zeichen akzeptiert. Ganz Hartgesottene können dieses Limit allerdings im Source-Code aushebeln und neu kompilieren.

12.6 Loggen über das Netz

Wenn Sie eine größere Zahl von Servern betreiben, ist es mühsam, sich jedes Mal dort einloggen zu müssen, wenn Sie die Log-Dateien kontrollieren möchten. In diesem Abschnitt lernen Sie, gezielt die gewünschten Protokolldaten auf einen anderen Server umzuleiten.

12.6.1 SyslogD

SyslogD empfängt Meldungen über das Netzwerk, wenn er mit dem Kommandozeilenparameter `-r` gestartet wird. Damit das bei jedem Start der Fall ist, wird in der Datei `/etc/default/syslogd` die Zeile

```
SYSLOGD="-r"
```


hinzugefügt. Nach einem Restart ist der Daemon auf dem UDP-Port 514 empfangsbereit. TCP wird nicht unterstützt, dies bleibt Syslog-ng und Rsyslog vorbehalten. Oft wird der Eintrag noch zu

```
SYSLOGD="-r -s example.com"
```

ergänzt. Durch den zweiten Parameter behandelt SyslogD alle Server aus der angegebenen Domäne als lokal und schreibt nur den einfachen anstelle des vollqualifizierten Hostnamens ins Log. Sollen dagegen Nachrichten an einen entfernten Host geschickt werden, genügt in der `/etc/syslog.conf` ein Eintrag wie der folgende:

```
*,* @hostname
```

Anstelle von `hostname` kann an dieser Stelle auch ein vollqualifizierter Name oder eine IP-Adresse stehen.

12.6.2 Syslog-ng

Syslog-ng empfängt Meldungen aus dem Netz, wenn ein passendes `source`-Objekt definiert ist. Im einfachsten Fall sieht das Objekt so aus:

```
source s_udp {udp(); }
```

Das ist die verkürzte Schreibweise für:

```
source s_udp {udp(ip (0.0.0.0) port(514)); }
```

Der UDP-Port 514 ist ohnehin der Default für den Empfang von Syslog-Meldungen, und `ip(0.0.0.0)` bedeutet, dass Syslog-ng auf allen vorhandenen IP-Adressen lauscht, falls der Server über mehr als eine verfügt. Nach dem gleichen Muster kann Syslog-ng angewiesen werden, auch auf einem TCP-Port Nachrichten entgegenzunehmen:

```
source s_tcp {tcp(ip (0.0.0.0) port(514)); }
```

Der TCP-Port 514 ist eigentlich für die `rshell` reserviert. Diese wurde jedoch schon vor Jahren flächendeckend von SSH verdrängt, sodass der Port guten Gewissens für TCP-Syslogging missbraucht werden kann.

Um einem entfernten System Nachrichten zukommen zu lassen, definiert man in Syslog-ng einfach ein entsprechendes `destination`-Objekt:

```
destination d_udp { udp( "10.100.3.22" ); };
```

Soll ein anderer Port als 514 genutzt werden, lautet die Zeile:

```
destination d_udp { udp( "10.100.3.22" port (10514) ); };
```

Soll für den Transport TCP statt UDP genutzt werden, lautet die Zeile analog:

```
destination d_tcp { tcp( "10.100.3.22" port (514) ); };
```

12.6.3 Rsyslog

Ältere Versionen von Rsyslog bringen Sie genau wie den klassischen SyslogD durch die Kommandozeilenoption `-r` dazu, Nachrichten per UDP auf Port 514 zu empfangen. Zusätzlich verstehen sie den Parameter `-t514`, der auch den TCP-Port 514 für den Empfang aktiviert. Beide Parameter können kombiniert werden, und andere Ports als 514 sind natürlich ebenfalls möglich. Rsyslog ab Version 3 akzeptiert diese Parameter nur noch widerwillig (nämlich dann, wenn man ihn zusätzlich mit `-c0` in den *sysklogd*-Kompatibilitätsmodus schaltet). Besser und flexibler ist es, die gewünschten Funktionen in der Konfigurationsdatei zu aktivieren. Dazu werden zunächst die nötigen Module geladen. Danach aktivieren Sie den Nachrichtempfang sowohl über UDP wie auch über TCP auf dem Port 514:

```
$ModLoad imudp
$ModLoad imtcp
$InputUDPServerRun 514
$InputTCPServerRun 514
```

Das Versenden von Meldungen an einen entfernten Host wird genau wie im SyslogD konfiguriert:

```
*.* @hostname
```

Soll der Transport per TCP erfolgen, muss ein weiteres »@«-Zeichen her:

```
*.* @@hostname
```

12.7 Syslog in eine Datenbank schreiben

Leider bieten weder SyslogD noch der freie Zweig von Syslog-ng die Möglichkeit, direkt in eine Datenbank zu loggen. Lediglich der kommerzielle Zweig von Syslog-ng – der hier allerdings nicht behandelt wird – und Rsyslog sind dazu in der Lage. Schauen wir uns also an, wie Rsyslog das Problem löst.

12.7.1 Anlegen der Log-Datenbank

Für die folgenden Beispiele benutzen wir MariaDB als Datenbanksystem, aber PostgreSQL wird ebenfalls unterstützt und analog konfiguriert. Gehen wir davon aus, dass MariaDB bereits erfolgreich installiert ist. Sollte das noch nicht der Fall sein, lesen Sie bitte zunächst Kapitel 11, »Datenbank«.

Auch wenn Sie den Rsyslog-Daemon bereits installiert haben sollten, werden Sie für die Datenbankbindung noch einmal die Paketverwaltung Ihrer Distribution bemühen müssen, denn dieser Teil der Funktionalität ist bei fast allen Distributionen in ein separates Paket ausgelagert, das meist *rsyslog-mysql* bzw. *rsyslog-pgsql* oder sehr ähnlich heißt. Während

der Installation dieses Pakets werden Sie nach dem root-Kennwort Ihres Datenbanksystems gefragt, damit das Datenbankschema aufgebaut werden kann. Nach getaner Arbeit finden Sie in MariaDB eine Datenbank mit dem Namen »Syslog«. Wenn Ihnen die Namensgebung nicht gefällt oder Sie dem automatischen Installer schlicht misstrauen, können Sie das Datenbankschema auch von Hand anlegen. Sie benötigen dazu das SQL-Skript *createDB.sql*, in dem Sie die gewünschten Anpassungen hinsichtlich des Datenbanknamens etc. vornehmen können. Leider wird das Skript bei einigen Distributionen nicht mitgeliefert. In diesem Fall müssen Sie es sich aus dem Source-Paket von der Rsyslog-Webseite besorgen, bevor Sie mit

```
mysql -u root -pPasswort < ./createDB.sql
```

die Datenbank erzeugen. Als Nächstes legen Sie auf dem SQL-Prompt noch ein weiteres Benutzerkonto an. Hier heißt die Datenbank »Syslog«, der Benutzer »rsyslog« und das Passwort »geheim«:

```
grant ALL ON Syslog.* to rsyslog@localhost identified by 'geheim'; flush privileges;
```

Listing 12.32 Benutzerkonto in MariaDB anlegen

Damit sind die notwendigen Vorarbeiten abgeschlossen. Im nächsten Schritt weisen Sie Rsyslog an, in die Datenbank hinein zu loggen.

12.7.2 In die Datenbank loggen

Unter */etc/rsyslog.d/* wird, falls sie noch nicht vorhanden ist, die Datei *mysql.conf* angelegt. Sie kann zum Beispiel so aussehen:

```
$ModLoad ommysql
mail.* :ommysql:127.0.0.1,Syslog,rsyslog,geheim
```

In der ersten Zeile wird das für die Datenbankanbindung nötige Rsyslog-Modul aufgerufen. In der zweiten Zeile wird zunächst mit *mail.** definiert, dass alle Log-Meldungen der Mail-Facility in die Datenbank geschrieben werden sollen. Danach folgen, und zwar in dieser Reihenfolge,

1. der Name des Datenbank-Konnektor-Moduls,
2. der Name oder die IP-Adresse des Hosts, auf dem der DB-Server läuft,
3. der Name der Datenbank,
4. der DB-Benutzername und
5. dessen Passwort.

Der Modulname, hier *ommysql*, wird dabei von Doppelpunkten umschlossen, und die weiteren Parameter sind durch Kommata voneinander getrennt. Die Tabelle, in der die Nachrichten gespeichert werden, heißt *SystemEvents*. Ihre Struktur ist in Tabelle 12.8 abgebildet.

Field	Type	Null	Key	Default	Extra
ID	int(10) unsigned	NO	PRI	NULL	auto_increment
CustomerID	bigint(20)	YES		NULL	
ReceivedAt	datetime	YES		NULL	
DeviceReportedTime	datetime	YES		NULL	
Facility	smallint(6)	YES		NULL	
Priority	smallint(6)	YES		NULL	
FromHost	varchar(60)	YES		NULL	
Message	text	YES		NULL	
NTSeverity	int(11)	YES		NULL	
Importance	int(11)	YES		NULL	
EventSource	varchar(60)	YES		NULL	
EventUser	varchar(60)	YES		NULL	
EventCategory	int(11)	YES		NULL	
EventID	int(11)	YES		NULL	
EventBinaryData	text	YES		NULL	
MaxAvailable	int(11)	YES		NULL	
CurrUsage	int(11)	YES		NULL	
MinUsage	int(11)	YES		NULL	
MaxUsage	int(11)	YES		NULL	
InfoUnitID	int(11)	YES		NULL	
SysLogTag	varchar(60)	YES		NULL	
EventLogType	varchar(60)	YES		NULL	
GenericFileName	varchar(60)	YES		NULL	
SystemID	int(11)	YES		NULL	

Tabelle 12.8 Struktur der Tabelle »SystemEvents«

Nun liegen die Log-Daten warm und trocken in der Datenbank und warten darauf, dass Sie Abfragen formulieren. Auf der Kommandozeile können Sie der Datenbank SQL-Statements übergeben und erhalten die Antwort direkt auf die Konsole. Im folgenden Beispiel soll MariaDB alle Log-Nachrichten anzeigen, die von dem Host *mailserver2* kamen und in deren Nachrichtentext der Name *charly* vorkommt:

```
mysqlq -u rsyslog -pgeheim -D Syslog -s -e "SELECT Message FROM SystemEvents
WHERE FromHost='mailserver2' AND Message LIKE '%charly%' "
```

Listing 12.33 Datenbankabfrage

Da eine Syslog-Datenbank meist nicht zur langfristigen Datenhaltung angelegt wird, sondern eher der punktuellen Fehlersuche und der einfacheren Handhabbarkeit bei statistischen Auswertungen dient, wird sie regelmäßig geleert. Um die Datenbank aus unserem Beispiel täglich um Mitternacht zu leeren, genügt diese Zeile in der Crontab:

```
0 0 * * * /usr/bin/mysql -u rsyslog -pgeheim -D mailLog
-e "truncate table SystemEvents"
```

Listing 12.34 Datenbank per »cron« leeren lassen

12.8 Fazit

In diesem Kapitel haben Sie verschiedene Möglichkeiten und Dienste kennengelernt, mit denen Sie die Meldungen Ihrer Systeme sammeln, auswerten und verwalten können. Welchen dieser Dienste Sie verwenden, bleibt Ihnen überlassen. Wichtig ist nur, dass Sie sich überhaupt mit dieser Thematik befassen, um immer einen Überblick über den »Gesundheitszustand« Ihres Systems zu haben. Aber welchen dieser Dienste sollten Sie jetzt wählen?

Wie Sie an den von uns eingesetzten Distributionen sehen, geht der Weg ganz klar in Richtung *journald* zusammen mit *systemd*. Auch wenn Ihnen der Gedanke nicht gefällt, dass Informationen nicht mehr im ASCII-Format, sondern binär gespeichert werden, sollten Sie sich mit *journald* und *systemd* auseinandersetzen. Auf die Dauer werden alle Distributionen den vollständigen Wechsel vollziehen, und je früher Sie sich mit den beiden Diensten beschäftigen, umso einfacher wird die Umstellung für Sie.

Kapitel 13

Proxy-Server

In diesem Kapitel erfahren Sie, wie Sie die Webzugriffe Ihrer Anwender mit einem Proxy-Server effektiv schützen und kontrollieren – inklusive Benutzeranmeldung, Gruppenzugehörigkeitsprüfung, Antiviren-Scan, Content-Filter und gezielter Log-Auswertung.

Ein Proxy (von engl. *proxy representative*, dt. *Stellvertreter*) stellt im Wesentlichen nichts anderes dar als einen Stellvertreter für Dienste. Im allgemeinen Sprachgebrauch hat es sich eingeschlichen, dass mit einem Proxy meist ein HTTP/HTTPS-Proxy-Server, auch *Web-Proxy* genannt, gemeint ist. Dies ist so nicht korrekt, da durchaus auch andere Dienste proxyfähig sind, zum Beispiel *FTP*, *NTP*, *NNTP* oder *SMTP*.

13

13.1 Einführung des Stellvertreters

In diesem Kapitel erfahren Sie alles zum Web-Caching-Proxy *Squid*, der den De-facto-Standard für Web-Proxys darstellt. Der *Squid* ist ein Fork des Anfang der 90er-Jahre entstandenen *Harvest Cache Daemon* und verfolgt das Ziel, die Caching-Technologie weiterzuentwickeln.

Neben der reinen Funktionalität als Proxy – also der Aufgabe, einer Vielzahl von Benutzern einen Dienst zentral zur Verfügung zu stellen – ist die Kernkomponente des *Squid* das Caching, also Inhalte lokal vorzuhalten, um den Bandbreitenbedarf zu senken. Bandbreite war Anfang der 90er-Jahre nicht nur teuer, sondern vor allem auch nur eingeschränkt verfügbar. Durch die stetig wachsende Anzahl von Inhalten des sich rasant entwickelnden neuen Mediums *Internet* kam die Überlegung auf, statische Inhalte (Bilder, Texte etc.) von häufig aufgerufenen Internetseiten lokal auf einem Proxy zu speichern und diese nur bei Veränderungen neu zu laden, sodass die Inhalte nicht nur wesentlich seltener die Bandbreite belasteten, sondern auch eine schnellere Verfügbarkeit für den Endbenutzer geboten werden konnte.

Aber zurück zum Stellvertreten. Beim Internetzugang via NAT (*Network Address Translation*), wie es im heimischen DSL angewandt wird, werden die Pakete einfach blind durchgereicht. Der Web-Proxy hingegen ist in der Lage, die erhaltenen Informationen auch zu sondieren. Neben der Möglichkeit, Virenschecks oder ein sogenanntes *Content Filtering* (Inhaltsprüfung) durchzuführen, kann auch eine Benutzerauthentifizierung eingebunden werden.

Der Web-Proxy *Squid* ist ein komplett ausgestatteter HTTP/1.0-Proxy und kompatibler HTTP/1.1-Proxy. Er wurde früher in vielen stabilen Versionen zur Verfügung gestellt, derzeit aber nur noch in folgenden:

- ▶ *Squid 5 (aktuell 5.7)*
- ▶ *Squid 6 (Development)*

Dies hat den Hintergrund, dass das Entwicklerteam mit dem bisherigen Konzept an die Grenzen des Machbaren gestoßen ist und beschlossen hat, den gesamten Quellcode auf einer zeitgemäßen Umgebung vollständig neu zu entwickeln.

Daher wurde die Version 3 in C++ geschrieben. Historisch gewachsene Überbleibsel wurden endgültig entfernt, und neue Techniken werden besser unterstützt. Version 4 führte dies fort und setzte einen C++-Compiler der Version 11 voraus. Ab Version 6 wird ein C++-Compiler der Version 17 benötigt.



Durch die Neuentwicklung sind die Konfigurationen teilweise nicht kompatibel. Ein Großteil ist zwar identisch, aber viele Kleinigkeiten wurden aktualisiert oder entfernt. Dankenswerterweise ist der Squid recht kommunikativ, sodass wir meist direkt beim Starten des Dienstes auf unseren Konfigurationsfehler hingewiesen werden.



Squid: Mindestens Version 4.6!

Wir empfehlen Ihnen, zur Vermeidung von Fehlern mindestens die Version 4.6 einzusetzen.

Auf der Projektseite www.squid-cache.org finden Sie eine umfangreiche Dokumentation aller Parameter und Verarbeitungsmethoden des Squid.

13.2 Proxys in Zeiten des Breitbandinternets

Sowohl durch den Ausbau von Verfügbarkeit und Geschwindigkeit der Internetzugänge als auch durch die neuen Techniken der Webpräsenzen (Java, Flash etc.) ist das Caching immer mehr in den Hintergrund geraten. Inhalte sollen und wollen nicht nur in Echtzeit präsentiert werden, sondern bestmöglich auch noch zu 100 % personalisiert – willkommen im Web 2.0. Dies widerspricht natürlich der Logik, Inhalte an einer Stelle für eine Vielzahl von Benutzern vorhalten zu wollen. Nichtsdestotrotz kann ein Web-Proxy nach wie vor den Bandbreitenbedarf verringern und somit eine bessere und schnellere Nutzung ermöglichen.

Der Sicherheitsgedanke spielt natürlich ebenfalls eine große Rolle. Durch den Betrieb eines Proxy-Servers müssen die Clients keine direkte Internetkommunikation aufbauen. Darüber hinaus gibt der Squid Ihnen Möglichkeiten zur Authentifizierung, Zugriffsregulierung und Zugriffsüberwachung an die Hand, wodurch Sie die Sicherheit weiter erhöhen können.

13.3 Herangehensweisen und Vorüberlegungen

Bevor Sie anfangen, einen Proxy-Server zu installieren und zu konfigurieren, müssen Sie ein paar Dinge bedenken. Das Web-Caching ist eine hohe Belastung für Festplatten, da sehr viele kleine Dateien ständig gelesen und geschrieben werden müssen. Daher sind schnelle Festplatten (am besten in einem RAID-Verbund) für einen Web-Proxy fast unerlässlich. Denn was nützt die geschonte Bandbreite, wenn die Geschwindigkeit aufgrund der zu hohen I/O-Last auf dem Proxy-Server einbricht? Ebenso müssen Sie beachten, dass ein Web-Proxy viel Hauptspeicher benötigt, um effizient arbeiten zu können. Hingegen wird die CPU des Systems durch den reinen Betrieb eines Web-Proxys nicht wirklich gefordert. Anders sieht es nun wieder aus, wenn wir zusätzlich noch einen Virenschanner oder Content-Filter betreiben wollen. Aus diesen Überlegungen folgt unweigerlich die Frage: »Wie muss mein Proxy-Server denn nun bestückt sein?«

Leider gibt es auf diese Frage keine pauschale Antwort, da die entsprechenden Werte nicht nur von der Anzahl der Nutzer und vom Nutzerverhalten an sich, sondern auch von den Performance-Anforderungen abhängen.

Der Web-Proxy eines Unternehmens mit 1.000 Angestellten muss nicht zwingend ein mit Hardware vollgestopft System sein. Wenn diese Benutzer lediglich eine Handvoll Internetseiten aufrufen dürfen, ist die Belastung des Systems relativ gering. Wenn aber hingegen ein Kleinbetrieb, zum Beispiel ein Softwareentwickler mit zehn Mitarbeitern, Vollzugriff auf das Internet hat und ständig Inhalte geladen werden müssen, dann sind die Hardwareanforderungen deutlich höher.

Darüber hinaus muss sich ein Administrator Gedanken über die Verfügbarkeit machen. Wenn Internetzugriffe geschäftsrelevant sind, kann ein Ausfall schnell teuer werden. Hier muss noch eine Besonderheit des Squid erwähnt werden: Skalierbarkeit ist einer der großen Vorteile des Squid, er ist nämlich von Haus aus fähig, eine verteilte Struktur zu betreiben. Im Squid-Jargon spricht man von sogenannten *Siblings*, *Childs* und *Parents*. Ein Sibling stellt einen gleichberechtigten Partner dar, mit dem via ICP (*Internet Cache Protocol*) Cache-Objekte ausgetauscht werden. Hingegen stellt ein *Child* nur eine Art Unter-Proxy dar, der seine Inhalte nicht direkt, sondern über einen *Parent*-Proxy bezieht.

13.4 Grundkonfiguration

Nach der Installation des Squid aus den Paketquellen Ihrer Distribution ist bereits eine Grundkonfiguration unter `/etc/squid/squid.conf` vorhanden – außer unter Debian: Dort müssen Sie hinter allen Pfaden stets die Zahl »3« ergänzen. Hier werden alle Parameter des Squid aufgeführt, und mit Kommentaren wird deren Funktionsweise erläutert. In diesem Abschnitt werden die Kernpunkte der Konfiguration und die Handhabung des Dienstes beschrieben. Folgende Punkte werden dabei näher betrachtet:



- ▶ Netzwerk
- ▶ Cache
- ▶ Logging
- ▶ Handhabung des Dienstes

Der komplexeste Konfigurationspunkt, das Regelwerk, wird aufgrund seines Umfangs in den beiden Abschnitten 13.4.6, »Objekte«, und 13.4.9, »Regeln«, im Detail erläutert.

13.4.1 Aufbau des Testumfelds

Für alle Konfigurationen in diesem Kapitel verwenden wir den Aufbau aus Abbildung 13.1.

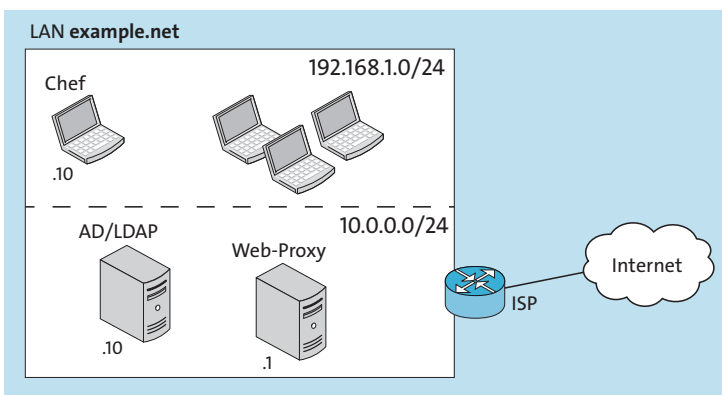


Abbildung 13.1 Netzwerkübersicht: Testumfeld Proxy

13.4.2 Netzwerk

Dem Dienst werden in der *squid.conf* über den Parameter *http_port* eine Netzwerkkarte und ein TCP-Port zugewiesen. Nach der Installation ist dieser Parameter so konfiguriert, dass der Squid auf allen Netzwerkschnittstellen auf dem TCP-Port 3128 lauscht:

```
# Squid normally listens to port 3128
http_port 3128
```

Listing 13.1 Portdefinition auf allen Netzwerkschnittstellen

Falls Sie dies einschränken wollen, sodass der Proxy nur auf einer Netzwerkschnittstelle läuft, fügen Sie die jeweilige IP-Adresse einfach hinter dem Befehl *http_port* hinzu, gefolgt von einem Doppelpunkt und der Portangabe:

```
http_port 10.0.0.1:3128
```

Listing 13.2 Portdefinition auf einer Netzwerkschnittstelle

13.4.3 Cache

Entsprechend der zur Verfügung stehenden Hardware können Sie dem Squid Hauptspeicher zuweisen. Dies geschieht über den Parameter `cache_mem`. In der Standardkonfiguration ist dieser bereits wie folgt definiert:

```
#Default:
# cache_mem 256 MB
```

Listing 13.3 Hauptspeicherkonfiguration

Den optimalen Wert für Ihr Umfeld finden Sie nur durch Erfahrung heraus, da er wieder sehr stark von der zur Verfügung stehenden Hardware und dem Benutzerverhalten abhängt. Als Grundlage sollten Sie mit einem Viertel bis der Hälfte Ihrer Hauptspeichergroße beginnen. Beachten Sie, dass Ihr OS und der Squid selbst natürlich auch noch Hauptspeicherbedarf haben.

Zu großer »cache_mem«

Dieser Parameter umschreibt nicht den maximalen von Squid genutzten Hauptspeicher, sondern lediglich den zusätzlich zur Verfügung gestellten Hauptspeicher für Cache-Objekte. Dieses Limit kann bei Bedarf vom Squid auch überschritten werden! Beachten Sie, dass ein zu groß gewählter Wert schnell zum Swapping führt, was die Verarbeitungszeit deutlich verlangsamt.

Des Weiteren können Sie die Größe und den Ort des lokalen Festplatten-Caches über den Parameter `cache_dir` definieren. Per Default ist ein Cache bereits vordefiniert, aber nicht aktiviert, wie in Listing 13.4 dargestellt. Hier wird das Verfahren `ufs` zur Cache-Speicherung im Verzeichnis `/var/spool/squid` mit 100 MB Größe in 16 Hauptverzeichnissen mit jeweils 256 Unterverzeichnissen angewandt. Das `ufs`-Verfahren ist das altherwürdige Squid-Verfahren zur Speicherung des Caches. Wesentlich aktueller ist das Verfahren `aufs`, das ein Update des `ufs`-Verfahrens darstellt. Hierbei werden die Lese-/Schreibzugriffe asynchron durchgeführt, was zu einer Geschwindigkeitssteigerung führen kann. Selbstverständlich erlaubt der Squid Ihnen auch, mehrere Cache-Pools zu betreiben, sodass Sie effektiv Messwerte erheben können, um somit das beste Verfahren für Ihr Umfeld zu ermitteln.

```
# Uncomment and adjust the following to add a disk cache directory.
#cache_dir ufs /var/spool/squid 100 16 256
```

Listing 13.4 Ort und Umfang des Cache-Verzeichnisses

Definieren Sie darüber hinaus die Größe der Objekte im Hauptspeicher und die maximale Größe von Cache-Objekten auf der Festplatte. Diese Werte sind in der Standardkonfiguration eher verhalten gesetzt. Passen Sie diese an zeitgemäße Werte an.



Über den Parameter *maximum_object_size_in_memory* definieren Sie die maximale Größe von Objekten im Hauptspeicher:

```
#Default:  
# maximum_object_size_in_memory 512 KB
```

Listing 13.5 Objektgröße im Hauptspeicher

Ab Version 3.1 wurde dieser Wert auf 512 KB erhöht (vorher 8 KB), damit kleinere Bilddateien auch im Hauptspeicher gehalten werden können.

Der Parameter *maximum_object_size* gibt die maximale Größe von Objekten an, die auf der Festplatte gespeichert werden. Auf aktuellen Systemen mit entsprechenden Festplatten können Sie diesen Wert durchaus auf 800 MB erhöhen, sodass ein ISO-CD-Image auch aus dem Cache an Clients geliefert werden kann:

```
#Default:  
# maximum_object_size 4 MB  
maximum_object_size 800 MB
```

Listing 13.6 Größe von Objekten auf der Festplatte



Leider gibt es auch hier keine Faustregel für die Berechnung der perfekten Einstellung. Selbstverständlich können Sie das Caching vollkommen an Ihre Bedürfnisse anpassen, sodass Sie den maximalen Wirkungsgrad zwischen vorhandener Bandbreite, Hauptspeicher, Festplattengröße und dem Surfverhalten Ihrer Nutzer erzielen. Allerdings übersteigen die dazu notwendigen Erläuterungen die Möglichkeiten dieses Buches.

13.4.4 Logging

Neben der Konfigurationsdatei spielen die Log-Dateien des Squid eine zentrale Rolle. Sie dienen nicht nur zur Auswertung der Zugriffe über den Proxy, sondern vor allem zur Fehleranalyse und dazu, Informationen über den Betriebsstatus zu erhalten. Die Logs werden unter dem Standardpfad */var/log/squid* abgelegt. Der Squid stellt drei Log-Dateien zur Verfügung: *access.log*, *cache.log* und *store.log*. Im *access.log* werden alle Zugriffe über den Proxy protokolliert. Jede Zeile repräsentiert einen Zugriff.

Da HTTP nicht auf Sessions beruht, werden beim Aufruf einer Webseite mehrere Zugriffe getätigt, die entsprechend protokolliert werden. So wird beim Aufruf einer Seite nicht nur die URL als solche protokolliert, sondern alle Seiten und Medien (zum Beispiel Bilder), die darin referenziert sind, einzeln für sich. Für den Endbenutzer erscheint der Vorgang zwar als ein Zugriff, in Wahrheit werden aber viele einzelne Dateien zu einem Gesamtbild zusammengefügt. Tabelle 13.1 zeigt die Daten, die bei jedem Zugriff im nativen Squid-Log-Format protokolliert werden.

Bezeichnung	Bedeutung
time	Timestamp des Zugriffs
elapsed	benötigte Zeit des Abrufens
remotehost	IP-Adresse des anfragenden Clients
code/status	Squid-Ergebnis/HTTP-Statuscode
bytes	Größe in Byte
method	verwendete Methode (GET, PUT etc.)
URL	die angeforderte URL
rfc931	Benutzername (bei erfolgreicher Authentifizierung)
peerstatus/peerhost	Abfragetyp (direkt oder über einen weiteren Proxy)
type	Typ der abgefragten URL (zum Beispiel <i>text/html</i>)

Tabelle 13.1 Elemente in der Datei »access.log«

Der Block *code/status* stellt das Ergebnis der Anfrage dar: zum einen das Squid-Ergebnis und zum anderen den HTTP-Statuscode. Das Squid-Ergebnis umschreibt, wie der Squid den Inhalt abgerufen hat oder warum dieser nicht abgerufen wurde. Tabelle 13.2 zeigt eine Übersicht über die gängigsten Squid-Ergebnisse und deren Bedeutung.

Code	Bedeutung
TCP_HIT	Inhalt ist im Cache vorhanden.
TCP_MISS	Inhalt ist nicht im Cache vorhanden.
TCP_REFRESH_MISS	Inhalt musste neu abgerufen werden (veralteter Cache-Inhalt).
TCP_REFRESH_HIT	Inhalt ist noch im Cache (kein erneuter Abruf).
TCP_CLIENT_REFRESH_MISS	Der Proxy wurde angewiesen, den Inhalt neu zu laden.
TCP_DENIED	Der Zugriff wurde untersagt.

Tabelle 13.2 Auszug aus den Squid-Ergebnissen

Zusätzlich werden die HTTP-Statuscodes protokolliert, die den Browserfehlermeldungen entsprechen. Tabelle 13.3 gibt einen Überblick über die gängigsten HTTP-Statuscodes.

Code	Wert	Bedeutung
200	OK	Die Abfrage verlief ohne Fehler.
301	Moved Permanently	Weiterleitung
304	Not Modified	Unverändert – Zustellung aus dem Cache
403	Forbidden	Zugriff untersagt
407	Proxy Authentication Required	Authentifizierung erforderlich
503	Service Unavailable	Der Inhalt kann zurzeit nicht abgerufen werden.

Tabelle 13.3 HTTP-Statuscodes

Selbstverständlich können die Squid-Fehlermeldungen angepasst werden. Über den Parameter `error_directory` können Sie das Verzeichnis und damit die Fehlermeldungen fixieren.

In der Grundkonfiguration wird die Sprache der Fehlermeldungen anhand der Browsereinstellungen des Clients gewählt (*multi-language support*).

```
# Deutsche Fehlermeldungen (openSUSE Leap/CentOS = /usr/share/squid/errors/de/):
# error_directory /usr/share/squid-langpack/de
```

Listing 13.7 Konfiguration der Fehlerausgabe

Falls Sie über eine ganzheitliche Lösung zur Log-Auswertung verfügen, können Sie das Format auch mittels `logformat` anpassen. Beim `cache.log` hingegen handelt es sich nicht etwa um ein Log der im Proxy-Cache abgelegten Daten, sondern um das Fehler-Log des Squid. In ihm wird zum Beispiel bei jedem `reload` und `restart` die Verarbeitungs- und Fehlerausgabe protokolliert. Im `store.log` hingegen werden die einzelnen Cache-Objekte protokolliert. Deaktivieren Sie dieses Log, da es unnötige I/O-Last erzeugt und für den Betrieb keine große Relevanz hat – es sei denn, Sie möchten ein voll angepasstes Caching etablieren. Falls Sie die Pfadangaben oder Dateinamen ändern wollen oder das entsprechende Log nicht anlegen lassen wollen, steht Ihnen das natürlich frei. Der Squid bietet hierfür die Parameter `access_log`, `cache_log` und `cache_store_log`, jeweils gefolgt von der Dateiangabe inklusive des Pfads oder der Option `none`, um das Log zu deaktivieren.

13.4.5 Handhabung des Dienstes

Sie starten den Dienst über die Konsole mittels `squid` im Daemon-Modus oder besser über `systemd` oder das `init.d`-Skript. Über die Systemmethoden können Sie den Dienst wie gewohnt starten, stoppen, neu starten und die Konfiguration neu laden. Änderungen in der

squid.conf können aber über zwei Methoden aktiviert werden: zum einen, wie bereits beschrieben, über das System selbst und zum anderen über den Squid-Parameter `reconfigure`:

```
root@web-proxy:~# squid -k reconfigure
```

Listing 13.8 Neuladen der Konfiguration

Welche Methode Sie benutzen, ist nur relevant, wenn Sie mehrere Instanzen von Squid gleichzeitig auf einem System betreiben wollen.

Das Neuladen der Konfiguration erfolgt unterbrechungsfrei, sodass Sie Änderungen auch im laufenden Betrieb ohne Aussetzer einbinden können. Dagegen hat das Neustarten des Dienstes – je nach Konfiguration des für den Squid zur Verfügung gestellten Speichers und der Festplattengeschwindigkeit – einen längeren Ausfall zur Folge. Dies beruht darauf, dass beim Neustart der gesamte Inhalt des im Hauptspeicher befindlichen Caches in Dateien geschrieben werden muss, was eine sehr hohe I/O-Last hervorruft.

Setzen Sie zusätzlich in der *squid.conf* den Benutzer und die Gruppe, unter dem bzw. der der Squid laufen soll. Jede Distribution liefert unterschiedliche Standards mit. Trotz dieser Standards sollten Sie den Benutzer und die Gruppe über die Parameter `cache_effective_user` und `cache_effective_group`, jeweils gefolgt von dem Benutzer bzw. der Gruppe, definieren. Dadurch haben Sie die Daten präsent, um den Dateien, auf die der Dienst zugreifen soll, die korrekten Rechte geben zu können.

Neben der reinen Konfiguration des Dienstes finden in der *squid.conf* auch die sogenannten ACLs (*Access Control Lists*) ihren Platz. Über sie werden die Authentifizierung und die Regulierung der Zugriffe realisiert. Die Definition erfolgt über Regeln und Objekte. Da diese die größten Stolpersteine bei der Konfiguration darstellen, werden wir uns in den nächsten beiden Abschnitten die Erstellung von und den Umgang mit Objekten und Regeln genauer ansehen.

Verwechslungsgefahr

Leider überschneiden sich beim Squid-Regelwerk die gängigen Bezeichnungen. Wird im Squid von einer `acl` gesprochen, so ist dort eigentlich die Objektdefinition gemeint.

Die Regel als solche wird hingegen mit `http_access` definiert. In diesem Buch werden nur die Begriffe »Objekt« und »Regel« verwendet, um einer Verwechslung vorzubeugen.



13.4.6 Objekte

Der Grundpfeiler einer jeden Regel sind Objekte. Sie umschreiben unter anderem Hosts, Netze, Ports oder Protokolle. Jedes Objekt enthält die in Tabelle 13.4 dargestellten Elemente.

Element	Bedeutung
acl	Einleitung der Objektdefinition
NAME	Name des Objekts, auf den innerhalb von Regeln verwiesen wird
TYPE	Typ des Objekts (<i>src, dst, port, proto, url_regex, dstdomain</i> etc.)
DATA	Daten des Typs (direkt oder in Dateien durch Anführungszeichen umschlossen)

Tabelle 13.4 Objektdetails

In der Standardkonfiguration sind bereits einige Objekte definiert:

```

acl localnet src 0.0.0.1-0.255.255.255 # RFC 1122 "this" network (LAN)
acl localnet src 10.0.0.0/8           # RFC 1918 local private network (LAN)
acl localnet src 100.64.0.0/10        # RFC 6598 shared address space (CGN)
acl localnet src 169.254.0.0/16       # RFC 3927 link-local (directly plugged) machines
acl localnet src 172.16.0.0/12        # RFC 1918 local private network (LAN)
acl localnet src 192.168.0.0/16       # RFC 1918 local private network (LAN)
acl localnet src fc00::/7             # RFC 4193 local private network range
acl localnet src fe80::/10            # RFC 4291 link-local (directly plugged) machines
acl SSL_ports port 443
acl Safe_ports port 80                # http
acl Safe_ports port 21                # ftp
acl Safe_ports port 443               # https
acl Safe_ports port 70                # gopher
acl Safe_ports port 210               # wais
acl Safe_ports port 1025-65535        # unregistered ports
acl Safe_ports port 280               # http-mgmt
acl Safe_ports port 488               # gss-http
acl Safe_ports port 591               # filemaker
acl Safe_ports port 777               # multiling http
# acl CONNECT method CONNECT         ### NUR bis squid Version 4 ! ###

```

Listing 13.9 Standardobjekte der »squid.conf«

Um einem Objekt mehrere Werte zuzuweisen, wird das Objekt mehrfach deklariert, zum Beispiel in Listing 13.9 `Safe_ports`. Oder die Werte werden in einer Zeile angegeben. Ebenfalls ist eine Mischform möglich, der Squid summiert die Auflistung stets zusammen.

In Objektdefinitionen werden keine *Wildcards* unterstützt. Der Squid bietet hierfür aber ein besonderes Domain-Handling an. Werden Domains mit einem führenden Punkt deklariert, dann sind in diesem Objekt auch alle Subdomains mit eingeschlossen. Bei der Angabe von `.example.net` sind die Domains `www.example.net` und `mail.example.net` enthalten. Zu-

dem sind Objektnamen *case-sensitive*, sodass das Objekt `linuxdistributionen` ungleich dem Objekt `LinuxDistributionen` ist.

Stolperstein Namensgebung

Jedes Objekt muss über einen eindeutigen Namen verfügen, den Sie in der `acl`-Zeile als zweiten Parameter angeben. Bei der Namenswahl müssen sich an die folgenden Spielregeln halten: Verwenden Sie nur Buchstaben, Zahlen, Binde- und Unterstriche. Leerzeichen, deutsche Umlaute oder Sonderzeichen sind nicht erlaubt und führen zu Fehlern!



13.4.7 Objekttypen

Der Squid unterscheidet Objekte nach Typen. Diese müssen Sie für jedes Objekt angeben, damit der Squid weiß, wie er das Objekt verarbeiten soll. Tabelle 13.5 zeigt eine Übersicht über die gängigsten Objekttypen.

13

Typ	Beispiel	Bedeutung
<code>src</code>	192.168.0.0/24	Quell-IP-Adresse, -Range oder -Netz
<code>dst</code>	192.168.1.0/30	Ziel-IP-Adresse, -Range oder -Netz
<code>srcdomain</code>	.example.com	Quellname
<code>dstdomain</code>	www.example.com	Zielname
<code>port</code>	8443	TCP-Port
<code>dstdom_regex</code>	s[0-9]*\.example\.com	regulärer Ausdruck auf Zieldomänen
<code>proxy_auth</code>	–	Authentifizierung

Tabelle 13.5 Objekttypen

13.4.8 Objektlisten in Dateien

Große Objektlisten innerhalb der Konfiguration zu pflegen ist nicht nur unübersichtlich, sondern stört zum Beispiel auch bei einer skriptgestützten Verarbeitung. Daher wurde die Möglichkeit geschaffen, Objektlisten in eine Datei auszulagern. Diese müssen nicht umständlich importiert werden. Hierfür genügt es, bei der Objektdefinition als Wert den Pfad zur Datei anzugeben und diesen mit doppelten Anführungszeichen (") zu umschließen. Wollen Sie zum Beispiel eine Liste der Windows-Update-Server in eine Datei auslagern, legen Sie zunächst die Textdatei `/etc/squid/windowsUpdate.txt` mit folgendem Inhalt an:

```
.windowsupdate.microsoft.com
.update.microsoft.com
.windowsupdate.com
.download.windowsupdate.com
wustat.windows.com
ntservicepack.microsoft.com
go.microsoft.com
dl.delivery.mp.microsoft.com
crl.microsoft.com
sls.microsoft.com
productactivation.one.microsoft.com
```

Listing 13.10 Externe »dstdomain«-Datei »/etc/squid/windowsUpdate.txt«

Damit der Squid die soeben angelegte Datei auch verarbeiten kann, müssen Sie die Rechte anpassen. Dies können Sie mit `chown proxy:proxy windowsUpdate.txt` erreichen. Anschließend deklarieren wir das Objekt `windowsUpdate` in der `squid.conf`:

```
acl windowsUpdate dstdomain "/etc/squid/windowsUpdate.txt"
```

Listing 13.11 Objektdefinition der externen »dstdomain«-Datei »windowsUpdate«

Somit wird der Squid angewiesen, den Inhalt aus der Datei `/etc/squid/windowsUpdate.txt` als Objekte des Typs `dstdomain` zu verarbeiten. Beachten Sie dabei, dass in der Datei pro Zeile nur ein Wert aufgeführt sein darf.

13.4.9 Regeln

Regeln stellen den größten Stolperstein für beginnende Proxy-Administratoren dar. Daher werden wir uns diese im Detail ansehen. Regeln erlauben oder verweigern nicht nur den Zugriff, sondern können diesen auch einschränken. Wie bei den Objekten sind in der Standardkonfiguration bereits Regeln definiert:

```
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost manager
http_access deny manager
http_access allow localhost
http_access deny all
```

Listing 13.12 Standardregeln

Eine Regel liest sich von links nach rechts. In Pseudocode lautet die Regel `http_access deny CONNECT !SSL_ports`: »Verbiete Zugriff auf das Objekt `CONNECT` (Zugriffsmethode `CONNECT`) für Nicht-Inhalte des Objekts `SSL_ports`.« Was so viel bedeutet wie: »Verbiete alle Zugriffe auf

Nicht-SSL-Ports.« Das führende Ausrufezeichen (!) bei `!SSL_ports` negiert das Objekt. Dies ist ein zentrales Mittel bei der Regelerstellung. Im Detail enthält jede Regel die in Tabelle 13.6 dargestellten Elemente.

Element	Bedeutung
<code>http_access</code>	Einleitung einer Regel
AKTION	Regelart: erlauben/verweigern (<code>allow/deny</code>)
OBJEKT	Objekt, auf das die Aktion angewendet wird
(optional) OBJEKT	Objekt, das den Zugriff einschränkt

Tabelle 13.6 Beschreibung von »`http_access`«

Regeln verfügen noch über ein paar Besonderheiten, die wir nun genauer betrachten. Da das Regelwerk sequenziell abgearbeitet wird, ist die Positionierung einer Regel von essenzieller Bedeutung. Es gilt der Grundsatz: »Die Reihenfolge ergibt sich vom Objekt mit den geringsten Rechten hin zu dem Objekt mit den meisten Rechten.« Darauf werden wir im weiteren Verlauf dieses Kapitels mehrfach eingehen.

Da der Squid das Prinzip des `first match` anwendet (Abbruch der Verarbeitung beim ersten Zutreffen einer Regel), sollte am Ende des Regelwerks immer eine `catch all`-Regel stehen, die alle Anfragen abfängt, die nicht bereits durch eine deklarierte Regel verarbeitet wurden. In der Standardkonfiguration ist dies bereits enthalten, wie in Listing 13.13 dargestellt:

```
# And finally deny all other access to this proxy
http_access deny all
```

Listing 13.13 Regel »`catch all`«

Betrachten wir in Tabelle 13.7 die zwei Methoden, um eine Freigabe zu definieren.

Beispiel	Regelwerk
A	<code>http_access allow myClients whitelisthttp_access deny all</code>
B	<code>http_access deny myClients !whitelisthttp_access deny all</code>

Tabelle 13.7 Zwei Arten, Freigaben zu definieren

Auf den ersten Blick erreichen beide Varianten das gleiche Ziel: Alle Mitglieder von `myClients` dürfen nur auf den Inhalt der `whitelist` zugreifen, und alle anderen Zugriffe werden verboten. Die beiden Beispiele werden von Squid aber unterschiedlich verarbeitet. Das Beispiel A arbeitet so, wie man es auf den ersten Blick erwartet. Beim Beispiel B hingegen werden

alle Zugriffe unterbunden. Dies ist darauf zurückzuführen, dass für das Objekt `myClients` lediglich `deny`-Regeln definiert sind und keine Regel, die einen Zugriff erlaubt.



Stolperstein

Falls keine Regel auf einen Zugriff zutrifft, wendet Squid das Gegenteil der letzten Regel in der Konfiguration an.

13.4.10 Überlagerung mit »first match«

Eine Feinheit der Regeln muss noch betrachtet werden: das Überlagern von Objektdefinitionen mithilfe von `first match`.



Den Mitarbeitern einer Firma ist der Zugriff auf Tageszeitungen untersagt, der Chef hingegen darf Boulevardzeitungen aufrufen. Dafür wurden folgende Objekte angelegt:

```
acl Chef src 192.168.0.10/32
acl Mitarbeiter src 192.168.0.0/24
acl Tageszeitungen dstdomain .bild.de
acl Tageszeitungen dstdomain .express.de
acl Tageszeitungen dstdomain .faz.de
acl Tageszeitungen dstdomain .waz.de
acl Boulevard dstdomain .bild.de
acl Boulevard dstdomain .express.de
```

Listing 13.14 Beispiel für eine Überlagerung: Objekte

Folgende Regeln müssten definiert werden, damit der Chef trotz der Überlagerung die Webseiten aufrufen darf:

```
http_access allow Chef Boulevard
http_access allow Mitarbeiter !Tageszeitungen
http_access deny all
```

Listing 13.15 Beispiel für eine Überlagerung: Regeln

Somit darf der Chef alle im Objekt `Boulevard` deklarierten URLs aufrufen. Allerdings würde auch dem Chef der Zugriff auf `www.waz.de` oder `www.faz.de` untersagt werden, da er durch die Class-C-Netz-Definition des Objekts `Mitarbeiter` ebenso Mitglied dieser Gruppe ist. Fassen wir das erworbene Wissen über Objekte und Regeln noch einmal zusammen:

- ▶ Objekte werden über `acl` definiert.
- ▶ Objektnamen sind *case-sensitive*.
- ▶ Objekte können in externe Dateien ausgelagert werden.

- ▶ Objekte werden horizontal und vertikal UND-verknüpft.
- ▶ Regeln werden über `http_access` definiert.
- ▶ Negierungen von Objekten in einer Regel werden mit einem führenden Ausrufezeichen (!) eingeleitet.
- ▶ Regeln werden sequenziell nach dem Prinzip des *first match* verarbeitet.

13.4.11 Anwendung von Objekten und Regeln

Um den Clients aus dem Beispiel, das in Abbildung 13.1 dargestellt ist, Vollzugriff über den Proxy zu gewähren, muss zuerst ein Objekt definiert werden:

```
acl myClients src 192.168.1.0/24
```

Listing 13.16 Objektdefinition »myClients«

Mit diesem Objekt wird nun eine Regel oberhalb des bestehenden Regelblocks erstellt:

```
http_access allow myClients
```

Listing 13.17 Regel »myClients«

Nach einem `reload` sind alle Clients des Netzes 192.168.1.0/24 in der Lage, Webseiten über den Proxy abzurufen. Wie bereits erwähnt wurde, kann der Zugriff auch eingeschränkt werden. Da Server im Normalfall lediglich Zugriff auf Update-Server benötigen, genügt es, zum Beispiel einem Ubuntu-Server Zugriff auf die Seiten *de.archive.ubuntu.com* und *security.ubuntu.com* zu gewähren. Dafür deklarieren wir ein neues Objekt `ubuntuUpdate` für die URLs und ein Objekt `myServer` für das Servernetz 10.0.0.0/24:

```
acl ubuntuUpdate dstdomain de.archive.ubuntu.com
acl ubuntuUpdate dstdomain security.ubuntu.com
acl myServer src 10.0.0.0/24
```

Listing 13.18 Objektdefinition »ubuntuUpdate« und »myServer«

Da für Windows-Updates deutlich mehr URLs deklariert werden müssen, empfiehlt es sich, die Auflistung in einer externen Datei vorzunehmen. Dafür legen wir die Textdatei */etc/squid/windowsUpdate.txt* mit folgendem Inhalt an:

```
.windowsupdate.microsoft.com
.update.microsoft.com
.windowsupdate.com
.download.windowsupdate.com
wustat.windows.com
ntservicepack.microsoft.com
go.microsoft.com
dl.delivery.mp.microsoft.com
```

```
curl.microsoft.com
sls.microsoft.com
productactivation.one.microsoft.com
```

Listing 13.19 Externe »dstdomain«-Datei »windowsUpdate.txt«

Beachten Sie, dass der Benutzer, unter dem der Squid-Daemon läuft, Leserechte auf diese Datei besitzt. Anschließend deklarieren wir das Objekt `windowsUpdate` in der `squid.conf`:

```
acl windowsUpdate dstdomain "/etc/squid/windowsUpdate.txt"
```

Listing 13.20 Objektdefinition der externen »dstdomain«-Datei »windowsUpdate«

Damit unsere Server im 10.0.0.0/24-Netz ausschließlich diese URLs aufrufen dürfen, muss noch eine entsprechende Regel erstellt werden. Dies sieht dann wie folgt aus:

```
http_access allow myClients
http_access allow myServer ubuntuUpdate
http_access allow myServer windowsUpdate
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
```

```
http_access deny all
```

Listing 13.21 Gesamtregelwerk



Gefahrenquelle HTTPS-Zugriffe

Die Konfiguration in Listing 13.21 birgt noch eine Gefahr in sich. Da hier lediglich *Whitelists* definiert wurden und nur eine *Catch-all*-Regel am Ende definiert wurde, könnten die Mitglieder des Objekts `myServer` auch HTTPS-Seiten aufrufen, die nicht in den Objekten `ubuntuUpdate` oder `windowsUpdate` enthalten sind – **first match!** Um dies zu verhindern, muss hierfür eine separate *Catch-all*-Regel vor der Regel `http_access deny !Safe_ports` eingefügt werden.

13.5 Authentifizierung

Eine Authentifizierung, also der Nachweis einer behaupteten Eigenschaft, wird von Squid als Zuordnung zu einer Gruppe behandelt. Zum einen kann die Zugehörigkeit zu solch einer Gruppe eine generelle Voraussetzung dafür sein, überhaupt den Proxy-Server verwenden zu dürfen. Zum anderen können diesen Gruppen auch dedizierte Rechte zugewiesen werden, sodass die Zugriffssteuerung noch feiner reguliert werden kann. Bei der einfachsten Form

der Authentifizierung handelt es sich um die *IP-basierte Authentifizierung*, die wir bereits in Abschnitt 13.4.10, »Überlagerung mit ›first match‹«, angewandt haben. Hier werden die Gruppen durch ihre IP-Adresse unterschieden. Das setzt natürlich eine feste Vergabe (oder Reservierung) der IP-Adressen voraus.

Der Squid sieht die IP-basierte Authentifizierung allerdings nicht als solche an, da diese Methode ohne eigentliche HTTP-Proxy-Authentifizierung auskommt. Eine *Proxy-Authentifizierung* erfolgt über den HTTP-Statuscode 407 – *Proxy Authentication Required*. Dieses Steuersignal weist den Browser an, die Authentifizierung durchzuführen, die dann meist ein Pop-up-Fenster im Browser auslöst, wie in Abbildung 13.2 dargestellt.

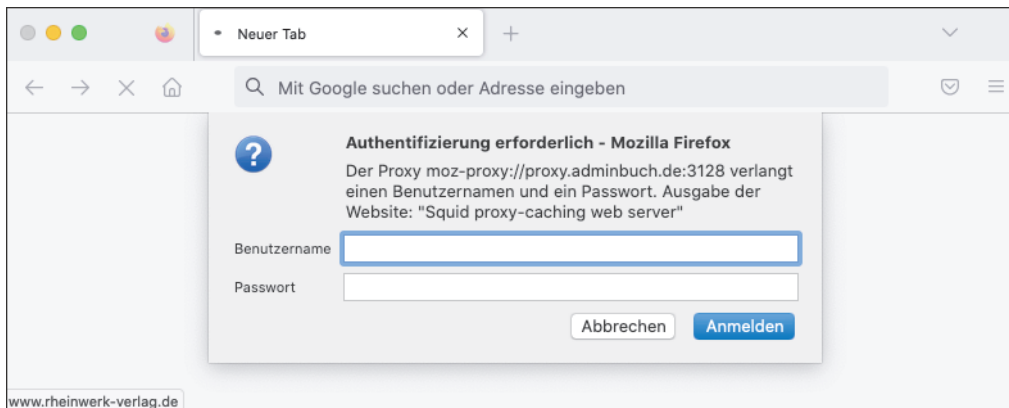


Abbildung 13.2 Proxy-Authentifizierung: Pop-up-Fenster

In diesem HTTP-Statuscode teilt der Proxy lediglich mit, welches Authentifizierungsschema verwendet wird. Der Proxy führt die Authentifizierung nicht selbst durch, sondern stellt dafür sogenannte *Helper* zur Verfügung. In nachstehender Auflistung sind die Helper aufgeführt, die von Squid angeboten werden:

- ▶ **DB**: Datenbank-Authentifizierung
- ▶ **LDAP**: LDAP-Verzeichnisdienst
- ▶ **NCSA**: Benutzer/Passwort-Datei im *htpasswd*-Format
- ▶ **MSNT**: Windows-NT-Authentifizierung
- ▶ **PAM**: Unix Pluggable Authentication Modules
- ▶ **SMB**: Windows NT oder Samba
- ▶ **radius**: Radius
- ▶ **getpwnam**: alte UNIX-Passwortdatei
- ▶ **SASL**: Simple Authentication and Security Layer
- ▶ **YP**: NIS-Datenbank

Zusätzlich werden noch *NTLM*, *Negotiate* (worunter auch Kerberos fällt) und *Digest* zur Verfügung gestellt, die im Gegensatz zu den gerade genannten Helfern die Passwörter verschlüsselt übertragen und somit eine höhere Sicherheit bieten.

Prüfen Sie, ob das von Ihnen favorisierte Authentifizierungsschema kompiliert wurde. Dies können Sie schnell über den Befehl `squid -v` herausfinden. Die Ausgabe des Befehls teilt Ihnen alle Kompilierungsoptionen mit.

Je nach verwendeter Methode werden die Daten an den Proxy übertragen. Authentifizierungsmethoden und deren Einstellungen werden in Squid mit dem Parameter *auth_param* beschrieben. Dieser Parameter wird mehrfach deklariert, um mehrere Einstellungen vorzunehmen.

Jede Methode hat dabei unterschiedliche Einstellungsmöglichkeiten. Tabelle 13.8 bietet eine Übersicht über die allgemeinen Parameter, die für alle Methoden gültig sind.

Parameter	Bedeutung
program	Helper-Datei (zum Beispiel <code>/usr/lib/squid/basic_ncsa_auth</code>)
children	Anzahl der gestarteten Prozesse

Tabelle 13.8 Authentifizierung: allgemeine Parameter

Da (einige) Authentifizierungsmethoden jeweils nur eine Anfrage parallel bearbeiten können, müssen mehrere Prozesse gestartet werden, damit im Falle von mehreren gleichzeitigen Authentifizierungen auch alle Anfragen bedient werden können.

Wählen Sie den Parameter `children` zu klein, so kommt es zu Verzögerungen, da der Squid auf die Abarbeitung warten muss.

Es können auch mehrere Authentifizierungsschemata in der *squid.conf* konfiguriert werden. Der Proxy stellt diese dann der Reihe nach dem Browser zur Verfügung. Dieser wählt dann das stärkste von ihm unterstützte Schema aus.

Die eigentliche Anweisung, eine Authentifizierung vorzunehmen, wird über die Zeile `acl <OBJEKTNAM> proxy_auth REQUIRED` beschrieben. Alle Helper liefern dem Squid nur zwei definierte Zustände zurück: zum einen »OK« bei erfolgreicher Authentifizierung und zum anderen »ERR« bei einer gescheiterten Authentifizierung.

Somit ist eine Unterscheidung, weshalb die Authentifizierung gescheitert ist (Rückgabewert »ERR«), nicht möglich. Durch diese einfache Struktur ist es aber wiederum möglich, eigene *Helper* zu programmieren. Im Squid-Wiki unter wiki.squid-cache.org finden Sie eine detaillierte Anleitung dazu.

13.5.1 Benutzerbasiert

Die Authentifizierung auf Benutzerebene hat natürlich große Vorteile gegenüber der IP-basierten. Zum einen können weniger privilegierte Nutzer nicht einfach über einen anderen Rechner (oder das Ändern ihrer eigenen IP-Adresse) an mehr Rechte gelangen, und zum anderen gelten die Freigaben von höher privilegierten Nutzern auch auf anderen Systemen als ihren eigenen.

In diesem Abschnitt widmen wir uns folgenden drei Szenarien:

- ▶ der lokalen Authentifizierung
- ▶ der Authentifizierung über Verzeichnisdienste
- ▶ der Erweiterung auf gruppenbasierte Authentifizierung

Wie bereits erläutert wurde, behandelt der Squid eine Authentifizierung als Zuordnung zu einer Gruppe. Dies können Sie sich zunutze machen, um eine Freigabe exakt an Ihre Bedürfnisse anzupassen.

Lokal

Die Authentifizierung gegen eine lokale Datei wird unter anderem über *basic_ncsa_auth* realisiert. Die Optionen `---enable-auth=basic` und `---enable-basic-auth-helpers=NCSA` müssen dafür beim Kompilieren gesetzt sein.

Eine *htpasswd*-Datei dient als Container der Benutzernamen und Passwörter. Damit die Benutzer des Testumfelds sich authentifizieren können, legen wir die Benutzer aus Tabelle 13.9 mithilfe von *htpasswd* an (siehe Listing 13.22).

Benutzer	Passwort
ben	blind
tom	deaf
jen	dumb

Tabelle 13.9 Testumfeld: Benutzer

```
root@web-proxy:~# htpasswd -c /etc/squid/users.passwd ben
```

```
New password: <PASSWORD>
```

```
Re-type new password: <PASSWORD>
```

```
Adding password for user ben
```

Listing 13.22 Anlegen einer »htpasswd«-Datei

Beachten Sie, dass der Schalter `-c` lediglich beim Anlegen verwendet wird. Da mit ihm eine neue Datei erstellt wird, würden Sie sonst die vorher angelegten Benutzer überschreiben. Anschließend kann diese Datei in die `squid.conf` eingebunden werden. Damit sie eingebunden werden kann, muss das Authentifizierungsschema vorher deklariert werden. In der Regel liegen diese Helper unter `/usr/lib/squid/`, einige Distributionen haben aber ein anderes Installationsverzeichnis für die *Helper*. Daher suchen Sie die Datei `basic_ncsa_auth` einfach mit `find / -name basic_ncsa_auth`. Nun kann dieses Schema deklariert und das Benutzerobjekt `myUsers` angelegt werden:

```
auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/users.passwd
acl myUsers proxy_auth REQUIRED
```

Listing 13.23 Authentifizierungsparameter »basic_ncsa_auth«

Anschließend entfernen wir die Freigabe für das Objekt `myClients` und fügen eine neue Freigabe unterhalb der Serverregeln für das neu angelegte Objekt `myUsers` ein. Komplette sieht die Regeldefinition nun wie folgt aus:

```
http_access allow myServer ubuntuUpdate
http_access allow myServer windowsUpdate
http_access allow myUsers
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access deny !myUsers
http_access deny all
```

Listing 13.24 Gesamtregelwerk

Die Reihenfolge ist wieder von essenzieller Wichtigkeit. Da Server Updates automatisiert abrufen, ist somit niemand vor Ort, der die Authentifizierung durchführen kann. Daher sollten die Server ohne vorherige Authentifizierung die benötigten Webseiten abrufen können. Bei der in Listing 13.24 aufgelisteten Reihenfolge wird dank des *first match* keine Authentifizierung von den Servern verlangt. Die Clean-up-Rule wurde um den Eintrag `http_access deny !myUsers` erweitert. Damit wird festgelegt, dass lediglich authentifizierte Benutzer zugelassen werden.

Verzeichnisdienst

Durch die Anbindung an einen Verzeichnisdienst ersparen Sie den Benutzern das erneute Einprägen von Anmeldeinformationen und uns Administratoren die doppelte Pflege. Am elegantesten lösen Sie dies durch die Benutzung eines *Single Sign-on*. Dabei werden die lokalen Anmeldedaten transparent an den Web-Proxy übertragen. Dies wird durch das

NTLM- und Kerberos-Authentifizierungsschema ermöglicht. Daher werden wir uns im Folgenden mit diesem Schemata auseinandersetzen. Zusätzlich werden wir die Authentifizierung mittels LDAP erläutern.

NTLM

Da die Authentifizierung nicht von Squid selbst durchgeführt wird, sondern Helper eingesetzt werden, muss das System in die Lage versetzt werden, die Verzeichnisdienste, also die Windows-Domäne, mittels NTLM abfragen zu können. Dies geschieht durch *Samba* und *Winbind*. Sorgen Sie dafür, indem Sie die entsprechenden Komponenten installieren und so konfigurieren, wie in Kapitel 15, »Samba4«, für Samba und Winbind beschrieben wird. Durch die Installation erhalten Sie einen weiteren Squid-Helper, der vom Samba-Team zur Verfügung gestellt wird. Diesen finden Sie unter `/usr/bin/ntlm_auth`.

Umbenennung der Squid-Helper

Früher hießen die Helper von Samba und Squid gleich. Ab Squid-Version 3.3 ist der Helper in `ntlm_smb_lm_auth` umbenannt worden, wodurch die Unterscheidung vereinfacht wurde.



13

Um diesen Helper verwenden zu können, müssen mindestens die Kompilierungsoptionen `---enable-auth=basic,ntlm` und `---enable-external-acl-helpers=wbininfo_group` gesetzt sein.

Sobald das System in der Lage ist, mit `wbininfo -u` die Benutzerinformationen des Verzeichnisdienstes auszulesen, kann mit `ntlm_auth` eine Authentifizierung vorgenommen werden. Da NTLM deutlich umfangreicher ist als eine Plain-Text-Authentifizierung, kann der Helper dementsprechend mehr Parameter verarbeiten. So kann er entweder die Authentifizierung via BASIC übertragen oder mit NTLM verschlüsselt durchführen. Dies machen wir uns zum Testen der Anbindung zunutze. Prüfen Sie mit dem nachstehenden Befehl, ob die Authentifizierung funktioniert:

```
root@web-proxy:~# ntlm_auth --helper-protocol=squid-2.5-basic
benutzername password
OK
```

Listing 13.25 Beispiel für BASIC-»ntlm_auth«

Irritierende Versionsangaben

Lassen Sie sich nicht durch die Angabe von `squid-2.5-basic` oder `squid-2.5-ntlmssp` irritieren. Die Bezeichnung ist lediglich historisch gewachsen und wurde aus Kompatibilitätsgründen nicht verändert.



Der verwendete Parameter `helper-protocol=squid-2.5-basic` leitet eine unverschlüsselte Übertragung ein. Diese Option für den Befehl `ntlm_auth` kann auch mit dem Parameter

squid-2.5-ntlmssp ausgeführt werden. Damit wird die Übergabe des Benutzernamens und des Passworts verschlüsselt erwartet.

Falls ein Client, der nicht an der Windows-Domäne angemeldet ist, über den Proxy Inhalte abrufen will, erscheint selbstverständlich ein Browser-Pop-up für die Authentifizierung. Hier besteht aber eine Besonderheit, da die Winbind-Konfiguration unterschiedlich sein kann. Falls in der *smb.conf* der Parameter `winbind separator = +` gesetzt ist, müssen die Anmeldeinformationen so wie in Abbildung 13.3 eingegeben werden.

Abbildung 13.3 Proxy-Authentifizierung bei »winbind separator = +«



Um Ihren Benutzern dies zu ersparen, setzen Sie den Parameter `winbind use default domain` in der *smb.conf* Ihres Samba-Servers auf `yes`. Damit ist zur Anmeldung die Angabe der Domäne nicht mehr erforderlich.

Konfigurieren Sie für Clients, die nicht in der Windows-Domäne angemeldet sind (zum Beispiel Linux-Desktops oder Gäste Ihres Hauses), ein *Fallback-Schema*, das dann zum Zuge kommt, wenn das erste Authentifizierungsschema gescheitert ist. Die Deklaration in der *squid.conf* könnte dann wie folgt aussehen:

```
auth_param ntlm program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-ntlmssp
auth_param ntlm children 10
auth_param basic program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-basic
auth_param basic children 10
auth_param basic realm Adminbuch Squid Web-Proxy
auth_param basic credentialsttl 2 hours
```

Listing 13.26 Beispiel für das Authentifizierungsschema NTLM: verschlüsselt und unverschlüsselt

Hier wird das Authentifizierungsschema NTLM zuerst in der verschlüsselten Variante angeboten und anschließend in der Klartextvariante. Zusätzlich wurde der Klartextvariante noch ein `realm` übergeben, der den angezeigten Text im Browser-Pop-up darstellt. Außerdem wurde noch die `credentialsttl` auf zwei Stunden gesetzt, was den Proxy veranlasst, die Prüfung auf Gültigkeit der übertragenen Anmeldeinformationen fortan nur alle zwei Stunden vorzunehmen. Leider ist dies in der verschlüsselten Variante nicht möglich, sodass jeder Zugriff eines Benutzers auch geprüft wird.

Ablaufen des Benutzerpassworts

Sollten Sie in Ihrer Domäne ein Intervall zum Ablauf der Gültigkeit von Passwörtern gesetzt haben, sodass sich Ihre Benutzer nach einer bestimmten Zeit selbst neue Passwörter geben müssen, kann ein Sonderfall eintreten. Da die Gültigkeit der Anmeldung auf Windows-Seite nur bei der Anmeldung geprüft wird, kann es passieren, dass ein Benutzer sich anmeldet und im Laufe des Tages die Gültigkeitsdauer überschritten wird. Geschieht dies, wird der Zugriff über den Proxy sofort unterbunden, da die Prüfung der Anmeldedaten pro Zugriff geschieht.

Da der Proxy nun Winbind zur Authentifizierung benutzt, muss er in die Lage versetzt werden, den Dienst auch nutzen zu können. Während der Authentifizierung wird auf die Datei `/var/run/samba/winbindd_privileged` zugegriffen. Daher muss der Benutzer, unter dem Squid läuft, auch entsprechend Rechte auf die Datei besitzen. Dies erreichen Sie, indem Sie den Benutzer, unter dem Squid läuft, mit in die Gruppe `winbindd_priv` aufnehmen. Dies erreichen Sie mit dem Befehl `usermod --a --G winbindd_priv squid`.

Rechtekorrektur

Verändern Sie bei der Datei `winbindd_privileged` lediglich die Gruppenzugehörigkeit oder fügen Sie Nutzer der Gruppe `winbindd_priv` hinzu, da ansonsten die Verarbeitung von Winbind abgebrochen wird!

Kerberos

Da das Authentifizierungsschema *NTLM* nach und nach aus dem Windows-Portfolio entfernt wird, sollten Sie in modernen Windows-Umgebungen besser auf *Kerberos* setzen. Falls Sie einen eigenen Kerberos-Server betreiben (siehe Kapitel 14, »Kerberos«), sollten Sie ebenfalls diese Authentifizierungsmethoden verwenden, da die Übertragung der Daten mittels *negotiate* erfolgt und somit verschlüsselt ist. Die Authentifizierung mittels Kerberos wird über den Helper `negotiate_kerberos_auth` realisiert. Um diesen aber einsetzen zu können, müssen mindestens die Kompilierungsoptionen `---enable-auth=negotiate` und `---enable-negotiate-auth-helpers=negotiate_kerberos_auth` gesetzt sein.

Damit Ihr Web-Proxy Benutzer mittels Kerberos authentifizieren kann, benötigt er einen Principal. Dieser ermöglicht es dem Squid, eigenständig mit dem Kerberos-Server zu kommunizieren und somit die Anmeldedaten zu prüfen. Wie Sie einen Service-Principal auf Ihrem eigenen Kerberos-Server erzeugen, lesen Sie in Abschnitt 14.7, »Hosts und Dienste«.

Für die Authentifizierung über eine Windows-Domäne müssen Sie den Principal entweder auf dem Domain-Controller vorab erzeugen (mittels `ktpass.exe`) oder mit Samba direkt auf dem Proxy. Zum Anlegen wird ein Domänen-Administratorkonto benötigt.



Uhrzeit!

Bedenken Sie, dass Kerberos zeitabhängig ist. Am besten verwenden Sie einen NTP-Server, um alle Zeitdaten gleich zu halten. Bei Abweichungen größer als 60 Sekunden kann keine Authentifizierung mittels Kerberos mehr durchgeführt werden.

Kerberos ist neben der Zeit auch von DNS abhängig. Richten Sie für Ihren Web-Proxy neben dem *A Resource Record*¹ in der *Forward Lookup*-Zone unbedingt auch einen *Pointer Resource Record*² in der *Reverse Lookup*-Zone ein. Nach diesen Vorarbeiten können Sie mit der eigentlichen Konfiguration beginnen.

Installieren Sie zunächst die Pakete *krb5-config*, *krb5-clients*, *krb5-user* und *samba-client*. Passen Sie die Konfigurationsdatei */etc/krb5.conf* Ihren Gegebenheiten an. Die in Listing 13.27 gezeigte Mindestkonfiguration sollte enthalten sein:

```
[libdefaults]
    default_realm = EXAMPLE.NET
    kdc_timesync = 1
    ccache_type = 4
    forwardable = true
    proxiable = true
    fcc-mit-ticketflags = true
    default_keytab_name = FILE:/etc/krb5.keytab

[realms]
    example.net = {
        kdc = ad.example.net
        master_kdc = ad.example.net
        admin_server = ad.example.net
        default_domain = example.net
    }

[domain_realm]
    .example = EXAMPLE.NET
    example = EXAMPLE.NET
```

Listing 13.27 Mindestkonfiguration von »/etc/krb5.conf«

Anschließend können Sie sich mittels Kerberos direkt an der Windows-Domäne authentifizieren und gleichzeitig das System für die weitere Konfiguration in Kerberos initialisieren:

1 *A Resource Record*, engl. für *Adresseintrag* = Zuordnung des DNS-Namens zur IP-Adresse.

2 *Pointer Resource Record*, engl. für *Adresszeiger* = Zuordnung der IPv4-Adresse zum Namen (*Reverse*).

```
root@web-proxy:/# kinit Administrator
Password for Administrator@EXAMPLE.NET:
```

Listing 13.28 Kerberos-Initialisierung mittels »kinit«

Dem Befehl *kinit* wird ein Domänen-Administrator übergeben. Nach erfolgreicher Authentifizierung wird Ihnen ein TGT zugewiesen. Dies können Sie mittels *klist* überprüfen:

```
root@web-proxy:/# klsit
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: Administrator@EXAMPLE.NET
```

```
Valid starting      Expires            Service principal
30.12.2022 17:34:15 31.12.2022 03:34:14 krbtgt/EXAMPLE.NET@EXAMPLE.NET
                renew until 31.12.2022 17:34:15
```

Listing 13.29 TGT-Anzeige mittels »klist«

Konfigurieren Sie nun den Samba-Server. Dabei ist ebenfalls nur eine Minimalkonfiguration, wie in Listing 13.30 aufgeführt, notwendig:

```
[global]
netbios name = example
realm = EXAMPLE.NET
security = ADS
encrypt passwords = yes
password server = ad.example.net
```

Listing 13.30 Samba-Konfiguration »smb.conf«

Anschließend können Sie Ihren Web-Proxy der Windows-Domäne hinzufügen:

```
root@web-proxy:/# net ads join -U Administrator
Enter Administrator's password:
Using short domain name -- EXAMPLE
Joined 'WEB-PROXY' to realm 'example.net'
```

Listing 13.31 Computerkonto für den Web-Proxy erstellen

Nun können Sie den eigentlichen Principal mit *net* erstellen:

```
root@web-proxy:/# export KRB5_KTNAME=FILE:/etc/squid/HTTP.keytab
root@web-proxy:/# net ads keytab add HTTP -U Administrator
```

```
Warning: "kerberos method" must be set to a keytab method to use keytab functions.
Processing principals to add...
Enter Administrator's password:
```

Listing 13.32 »keytab« für den Web-Proxy erzeugen

Principals werden in *keytab*-Dateien gespeichert. Falls Sie mehrere Dienste über Kerberos absichern möchten, sollten Sie stets eigene *keytab*-Dateien für jeden Dienst verwenden. Den Speicherort gibt die Umgebungsvariable `KRB5_KTNAME` an. Um zu prüfen, ob der Principal auch erstellt wurde, können Sie das Programm `ktutil` verwenden:

```
root@web-proxy:/# ktutil
ktutil: rkt /etc/squid/HTTP.keytab
ktutil: list
slot KVNO Principal
-----
 1  2 HTTP/web-proxy.example.net@EXAMPLE.NET
 2  2 HTTP/web-proxy.example.net@EXAMPLE.NET
 3  2 HTTP/web-proxy.example.net@EXAMPLE.NET
 4  2      HTTP/web-proxy@EXAMPLE.NET
 5  2      HTTP/web-proxy@EXAMPLE.NET
 6  2      HTTP/web-proxy@EXAMPLE.NET
ktutil: quit
```

Listing 13.33 Inhalt der »HTTP.keytab« anzeigen



Stolperstein: »FQDN«

Achten Sie darauf, dass auch für den FQDN Ihres Web-Proxys ein Principal hinterlegt ist. Fehlt dieser, kann die Authentifizierung gestört sein! Prüfen Sie zunächst die Einträge in Ihrem DNS und die lokale *resolv.conf*.

Damit der Squid mit *keytab*-Dateien arbeiten kann, müssen Sie die Rechte anpassen:

```
root@web-proxy:/# chmod 740 /etc/squid/HTTP.keytab
root@web-proxy:/# chgrp proxy /etc/squid/HTTP.keytab
```

Listing 13.34 Rechtekorrektur für »HTTP.keytab«

Achten Sie darauf, dass Sie als Gruppenzugehörigkeit die in Ihrer *squid.conf* angegebene Gruppe verwenden.

Damit der Squid nicht die Standard-*keytab*-Datei verwendet, sondern die extra für ihn angelegte *HTTP.keytab*, sollten Sie die *init.d*-Datei um die Umgebungsvariable `KRB5_KTNAME` erweitern. Fügen Sie dafür einfach in der Sektion *start* die Zeilen aus Listing 13.35 hinzu. Details zum Thema Kerberos finden Sie in Kapitel 14, »Kerberos«.

```
KRB5_KTNAME=/etc/squid/HTTP.keytab
export KRB5_KTNAME
```

Listing 13.35 Anpassung von »/etc/init.d/squid«

Da Ihr Web-Proxy nun in der Lage ist, über Kerberos Benutzer zu authentifizieren, können Sie die Kerberos-Authentifizierung mit `auth_param` in Squid einbinden:

```
auth_param negotiate program /usr/lib/squid/negotiate_kerberos_auth -d
auth_param negotiate children 10
auth_param negotiate keep_alive on
```

Listing 13.36 Einbindung von Kerberos: »squid.conf«

Der Helper `negotiate_kerberos_auth` wird mit dem Parameter `-d` in den Debug-Modus versetzt. Entsprechend länger sind die Ausgaben im `cache.log`. Nach einem erfolgreichen Test können Sie diesen Parameter wieder entfernen.

Ein *Fallback-Schema* gibt es bei Kerberos nicht, da dies nur verschlüsselt stattfinden kann. Im Falle einer Windows-Domäne findet als *Fallback-Schema* meist LDAP Anwendung. Wie Sie dieses konfigurieren, erfahren Sie im folgenden Abschnitt.

Stolperstein: Client-Proxy-Konfiguration

Achten Sie darauf, dass Ihr Web-Proxy bei den Clients mit seinem Namen im Browser eingetragen ist und nicht mit seiner IP-Adresse! Neben der Abhängigkeit von der korrekten Zeit ist Kerberos ebenfalls von DNS abhängig.



13

basic_ldap_auth

Falls Sie einen eigenen LDAP-Server betreiben (siehe Kapitel 17, »LDAP«) oder falls in einem Windows-AD das LDAP aktiviert ist, dann können Sie selbstverständlich auch darüber Benutzer authentifizieren. Dafür sind keine Vorarbeiten am System notwendig. Sie müssen lediglich Ihre Daten für den LDAP-Helper `basic_ldap_auth` anpassen und direkt in der `squid.conf` einrichten:

```
auth_param basic program /usr/lib/squid/basic_ldap_auth -v 3 \
-b "dc=example,dc=net" -D uid=squid-user,ou=Benutzer,dc=example,dc=net \
-w password -f sAMAccountName=%s ldap.example.net
auth_param basic children 5
auth_param basic realm Web-Proxy
auth_param basic credentialsttl 30 minute
```

Listing 13.37 Beispiel für das Authentifizierungsschema »ldap_auth«

Dabei haben die einzelnen Parameter folgende Bedeutung:

- ▶ **-v 3**: gibt die verwendete LDAP-Version an.
- ▶ **-b**: spezifiziert das *BaseDirectory* (Wurzelverzeichnis).
- ▶ **-D**: Benutzer, mit dem der Squid sich am LDAP authentifiziert

- ▶ **-w**: Passwort des Benutzers
- ▶ **-f**: Filter (Gibt an, gegen welches LDAP-Attribut authentifiziert werden soll.)
- ▶ **[LDAP-SERVER]**: gibt den LDAP-Server an.



Sicherheitsrisiko

Für den LDAP-Benutzer genügt im AD ein Gastkonto. Verwenden Sie nie ein Konto mit erhöhten Rechten oder sogar den LDAP-root-Benutzer (oder im AD einen Domänen-Administrator), da das Kennwort des Benutzers in der *squid.conf* im Klartext gespeichert wird!

Verwenden Sie als Filter (*-f*) im AD-Betrieb das Attribut *sAMAccountName*. Dieses stellt den Anmeldenamen des AD-Benutzers dar. Falls Sie die Authentifizierung an einen eigenen LDAP-Server richten möchten, können Sie selbstverständlich auch das Attribut *uid* oder *cn* verwenden. Falls Sie nicht wissen, wie die LDAP-Struktur im AD aufgebaut ist, können Sie diese auch mit `ldap_search` herausfinden.

13.5.2 Gruppenbasiert

Die weitere Unterteilung der Benutzer in Gruppen wird mit einem *external_helper* realisiert. Je nach verwendeter Authentifizierungsmethode können Sie die dazugehörigen Helper verwenden. Einige Methoden besitzen aber gar kein Gruppen-Pendant, so wie Kerberos: Dort kommt in den meisten Fällen LDAP zum Einsatz. Die Einbindung in Squid erfolgt über den Parameter *external_acl_type*. Über diesen Parameter können verschiedenste Variablen einem externen Helper zur Verfügung gestellt werden, der dann äquivalent zu `auth_param` das Ergebnis in den Zuständen *OK* und *ERR* zurückliefert.

Variable	Bedeutung
%LOGIN	Anmeldeinformationen
%SRC	Client-IP-Adresse
%DST	Ziel-IP-Adresse
%PROTO	abgefragtes Protokoll
%PATH	abgefragter URL-Pfad
%METHOD	abgefragte Methode (GET, POST, CONNECT etc.)
%PORT	abgefragter TCP-Port

Tabelle 13.10 Auszug aus Variablen für »external_acl_type«

Zusätzlich können Optionen gesetzt werden, um die Verarbeitung zu beeinflussen:

Option	Bedeutung
children	Anzahl der zu startenden Child-Prozesse
ttl	Dauer der Gültigkeit nach einer erfolgreichen Überprüfung (Default: 1 Std.)
negative_ttl	Dauer der Gültigkeit nach einer gescheiterten Überprüfung (Default: 1 Std.)
cache	Größe des Ergebnis-Caches (Default: unbegrenzt)

Tabelle 13.11 Auszug aus den Optionen von »external_acl_type«

NTLM: ext_wbinfo_group_acl

Für NTLM mit Winbind ist das Gruppen-Pendant *ext_wbinfo_group_acl*, ein Skript, das von Squid zur Verfügung gestellt wird. Dieses Perl-Skript arbeitet äquivalent zu *ntlm_auth*. Es prüft die durch ein Leerzeichen getrennten Benutzer- und Gruppennamen auf Gültigkeit. Es kann ebenfalls wieder in der Konsole getestet werden:

```
root@web-proxy:~# /usr/lib/squid/ext_wbinfo_group_acl
benutzername windows-ad-gruppe
OK
```

Listing 13.38 Prüfung der Gruppenzugehörigkeit via »ext_wbinfo_group_acl«

Damit können Sie prüfen, ob ein Benutzer einer bestimmten Gruppe angehört.

Für die Authentifizierung einer Gruppe verwenden wir die Variable %LOGIN. Fügen Sie in der Squid-Konfigurationsdatei daher folgende Zeile hinzu:

```
external_acl_type AD_Gruppe ttl=3600 children=5 %LOGIN \
/usr/lib/squid/ext_wbinfo_group_acl
```

Listing 13.39 Squid: »ext_wbinfo_group_acl«

Die Zeile gibt an, dass die Anmeldedaten (%LOGIN) dem externen Helper *ext_wbinfo_group_acl* übergeben werden. Durch die Option *children* wird die Anzahl der zu startenden Prozesse angegeben (im Beispiel fünf Prozesse). Die ermittelten Ergebnisse haben eine Gültigkeit von 3.600 Sekunden und werden im Objekt *AD_Gruppe* abgelegt.

Achtung: sequenzielle Verarbeitung – zu wenig Helper-Prozesse

Hier gilt ebenfalls, dass der Squid sequenziell arbeitet – also nach dem Motto: »Einer nach dem anderen!« Dadurch kann es zu Verzögerungen kommen, wenn Sie die Anzahl der zu startenden Prozesse zu klein gewählt haben.



Zusätzlich müssen Sie Gruppen mittels `acl` definieren:

```
acl Abteilung2 external AD_Gruppe Abt2
acl Abteilung3 external AD_Gruppe Abt3
acl Admins external AD_Gruppe Domänen_Admins
```

Listing 13.40 Squid: AD-Gruppendefinition

Der bei der Definition hinzugefügte Parameter `external` gibt an, dass das nachstehende Objekt einem `external_acl_type` angehört. Das vollständige Squid-Regelwerk sieht dann wie folgt aus:

```
external_acl_type AD_Gruppe ttl=3600 children=5 %LOGIN \
/usr/lib/squid/ext_wbinfo_group_acl

acl Whitelist dstdomain .example.net
acl Blacklist dstdomain .bild.de
acl Blacklist dstdomain .express.de
acl Blacklist dstdomain .faz.de
acl Blacklist dstdomain .waz.de
acl Abteilung2 external AD_Gruppe Abt2
acl Abteilung3 external AD_Gruppe Abt3
acl Admins external AD_Gruppe Domänen_Admins
acl myUsers proxy_auth REQUIRED
```

```
http_access allow Admins
http_access allow Abteilung2 !Blacklist
http_access allow Abteilung3 Whitelist
http_access deny !myUsers
http_access deny all
```

Listing 13.41 Squid: Regelwerk zur Authentifizierung mit Gruppenprüfung

LDAP: `ext_ldap_group_acl`

Bei einer LDAP- oder Kerberos-Authentifizierung bietet sich die Gruppenauthentifizierung mittels LDAP an. Dem entsprechenden Helper `ext_ldap_group_acl` muss das LDAP-Umfeld durch einen Parameter bekannt gemacht werden. Dies konfigurieren Sie, äquivalent zu `basic_ldap_auth`, direkt in der `squid.conf`:

```
external_acl_type AD_Group %LOGIN /usr/lib/squid/ext_ldap_group_acl -R \
-b "DC=example,DC=net" -h ldap.example.net -S \
-D "CN=<LDAP-USERNAME>,CN=Users,DC=example,DC=net" \
-w <PASSWORD> -f "(&(CN=%g)(member=%u)(objectClass=group))" \
-s sub -F "sAMAccountName=%s" -K
```

Listing 13.42 Konfiguration der LDAP-Gruppenprüfung: »squid.conf«

Die Parameter haben dabei die gleiche Bedeutung wie bei *basic_ldap_auth* (siehe Abschnitt 13.5.1, »Benutzerbasiert«). Die zusätzlichen Parameter dienen dazu, den Abruf zu beschleunigen. Den Hauptfokus müssen Sie allerdings auf den Filter (*-f*) legen. Dieser spezifiziert nämlich, wie die Gruppen abgefragt werden. In Listing 13.42 wird eine durch UND (&) verknüpfte Filterkette verwendet. Diese auf den ersten Blick verwirrenden Angaben lassen sich leicht entschlüsseln. Die einzelnen Glieder, die von Klammern umschlossen sind, filtern dabei folgende Attribute:

- ▶ **(CN=%g)**
Dies ist der Gruppenname.
- ▶ **(member=%u)**
Der Benutzer muss Mitglied der Gruppe sein.
- ▶ **(objectClass=group)**
Es muss sich um eine Gruppe handeln.

In Pseudocode ausgedrückt, würde die Filterkette Folgendes bedeuten: »Prüfe, ob der übergebene Gruppenname (%g) im LDAP existiert UND ob der Benutzer (%u) Mitglied dieser Gruppe ist, und das nur, wenn es sich beim LDAP-Element um eine Gruppe handelt.«

Die Variablen werden über das %LOGIN zur Verfügung gestellt und bestehen zum einen aus den Anmeldedaten des Benutzers und zum anderen aus den Gruppenangaben in den ACLs.

Je nachdem, wie Ihre LDAP-Struktur aufgebaut ist, müssen Sie den Filter anpassen, damit alle Ihre Benutzer auch authentifiziert werden können.

Da es sich hierbei um einen Klartext-Helper handelt, können Sie die Funktion auch auf der Konsole prüfen. Wie Sie den Filter aufbauen, bleibt ganz Ihnen überlassen, beachten Sie nur, dass der Web-Proxy in der Lage sein muss, alle Gruppen auch Benutzern zuzuweisen – falsche Filter können dazu führen, dass Ihr Web-Proxy keinen Zugriff erlaubt.

13.6 Log-Auswertung: Calamaris und Sarg

Eine Log-Analyse kann bei größeren Umgebungen schnell in unüberschaubare Arbeit ausarten. Damit Sie trotzdem den Überblick behalten, gibt es viele Tools, die Ihnen die mühsame Kleinarbeit abnehmen und aktuelle Werte kompakt für Sie darstellen.

13.6.1 Calamaris

Zu den ältesten und am weitesten verbreiteten Tools zählt *Calamaris*, das von Cord Berman geschrieben und unter der GPL veröffentlicht wurde. Das in Perl geschriebene Tool analysiert das Squid-*access.log* (oder auch jede Menge andere Log-Formate) und generiert einen umfangreichen Bericht. Nach der Installation aus den Paketquellen steht Ihnen das

Tool zur Verfügung. Die einfachste Möglichkeit, einen Calamaris-Report zu erzeugen, ist die folgende:

```
root@web-proxy:~# cat /var/log/squid/access.log | calamaris -a -F html > \
/var/www/calamaris/index.html
```

Listing 13.43 Einfacher Calamaris-Report



CentOS

Leider fehlt Calamaris in den Paketquellen von CentOS. Daher müssen Sie die Quellen von der Projektseite <https://cord.de/calamaris-english> herunterladen und es selbst installieren.

Ein wesentliches Merkmal von Calamaris besteht darin, dass Sie den Typ des Ausgabeformats definieren können (siehe Tabelle 13.12).

Bezeichnung	Bedeutung
mail	Fügt dem Report eine Betreffzeile hinzu.
html	HTML-formatierter Report, kann mit <i>mail</i> kombiniert werden.
html-frame	HTML-Report mit Frames und Tabellen
html-embed	HTML-Report ohne Header, um diesen in bestehende Webseiten einzufügen
graph	Fügt dem HTML-Report Graphen hinzu.
unformatted	Zahlenwerte werden unkodiert und durch Leerzeichen getrennt ausgegeben.

Tabelle 13.12 Calamaris-Ausgabeformate



Selbstverständlich verfügt Calamaris über eine Vielzahl an Parametern, mit deren Hilfe Sie den Report Ihren Bedürfnissen anpassen können. Benötigen Sie zum Beispiel keine Namensauflösung für den Report, sondern nur die Top-100-URLs, die eingehenden TCP-Verbindungen, die Anfragen nur ab der 2nd Domain und die Größenangaben in Megabyte, so könnte Ihr Calamaris-Aufruf wie folgt aussehen:

```
root@web-proxy:~# cat /var/log/squid/access.log | calamaris -n -d 100 \
-S 5,8 -F html -U M > /var/www/calamaris/index.html
```

Listing 13.44 Eingeschränkter »Calamaris«-Report

Bei der Installation werden bereits Einträge für die tägliche Log-Auswertung angelegt. Falls Sie dies lieber selbst steuern wollen, entfernen Sie die Datei */etc/cron.daily/calamaris*.

13.6.2 Sarg

Sarg (*Squid Analysis Report Generator*) erstellt einen benutzerbezogenen HTML-Report. Sarg wurde 1998 als *sqmgrlog* (*Squid Manager Log*) von Pedro Lineu Orso ins Leben gerufen und 2001 in Sarg umbenannt. Nach seiner Veröffentlichung fand das Projekt viel Zuspruch und erhielt Anregungen aus der Benutzerschaft, was die weitere Entwicklung deutlich geprägt hat. Nach der Installation finden Sie unter `/etc/sarg/` die `sarg.conf`, die die zentrale Konfigurationsdatei darstellt.

CentOS und Debian

Leider fehlt Sarg in den Paketquellen von CentOS und Debian. Daher müssen Sie die Quellen von der Projektseite <https://sourceforge.net/projects/sarg/> herunterladen und es selbst kompilieren.



Hier ein Auszug der gängigsten Parameter und ihre Bedeutung:

- ▶ `access_log /var/log/squid/access.log`
Ort des Squid-*access.log*
- ▶ `title`
Titel der HTML-Seite
- ▶ `output_dir /var/lib/sarg`
Pfadangabe des Reports. Passen Sie diese an Ihre Gegebenheiten an (beispielsweise `/var/www/sarg/`).
- ▶ `user_ip no`
Wenn Sie diesen Schalter auf `yes` setzen, werden die IP-Adressen anstatt der Benutzernamen in den Report geschrieben.
- ▶ `date_format u`
Setzen Sie diesen Schalter auf `e` für ein europäisches Datumsformat.
- ▶ `topsites_num 100`
Anzahl der Seiten in der Topsites-Übersicht
- ▶ `report_type`
Entfernen Sie die Typen, die Sie nicht in Ihren Reports benötigen.
- ▶ `#squidguard_conf /etc/squid/squidGuard.conf`
Speicherort der *squidGuard.conf*, falls Sarg die *squidGuard*-Logs mit auswerten soll

Wie auch bei Calamaris werden bei Sarg bereits während der Installation Einträge für die Log-Auswertung mittels Cron vorgenommen. Falls Sie dies selbst steuern wollen, entfernen Sie die Dateien `/etc/cron.daily/sarg`, `/etc/cron.weekly/sarg` und `/etc/cron.monthly/sarg`.

13.7 Unsichtbar: transparent proxy

In vielen Installationen ist eine Konfiguration der Proxy-Einstellungen im Browser undenkbar: zum Beispiel, wenn Sie reisende Mitarbeiter beschäftigen, die auch andere Internetzugänge als die Ihres Hauses benutzen müssen, aber nicht über die Rechte verfügen, die Einstellungen zu verändern; oder aber wenn Sie Ihren Kunden vor Ort schnell einen Webzugriff zur Verfügung stellen wollen, zum Beispiel über ein HotSpot-WLAN. Hierfür stellt Squid den sogenannten *transparenten Modus* zur Verfügung. Dieser setzt voraus, dass der Proxy als Gateway genutzt wird – dies kann tatsächlich so sein oder aber via *Policy Based Routing* (engl. für *regelbasiertes Routing*) realisiert werden. Für den Client stellt sich die Verbindung zum Webserver/Webdienst direkt dar, im Hintergrund ruft aber der Squid die Inhalte auf und reicht diese weiter. Trotz dieser Einschränkung ist ein transparenter Proxy oft das Mittel der Wahl. In der *squid.conf* wird dies über eine Erweiterung des Parameters `http_port` konfiguriert:

```
http_port 3128 intercept
```

Listing 13.45 Aktivierung des transparenten Modus



Änderung des Parameters!

In älteren Versionen (vor 3.0) wurde der transparente Modus mittels *transparent* eingeleitet!

Zusätzlich müssen die HTTP-Anfragen, die den Proxy erreichen, nun noch an den Squid weitergegeben werden, damit sie auch von ihm verarbeitet werden können. Hierfür müssen Sie die *iptables*-Regel aus Listing 13.46 hinzufügen:

```
iptables -t nat -A PREROUTING -i enp0s3 -p tcp --dport 80 -j REDIRECT --to-port 3128
```

Listing 13.46 »iptables«: Weiterleitung von TCP-Port 80 an Port 3128

Falls Sie weitere Webdienste über den Proxy anbieten wollen, zum Beispiel wenn ein Webdienst auf dem Port 8080 läuft, müssen Sie diesen ebenfalls via *iptables* umleiten.



Da HTTPS-Anfragen über Squid im transparenten Modus nicht verarbeitet werden können, müssen Sie diese direkt weiterleiten (NAT).

Mit einem Proxy als Default-Gateway sollte die vollständige *iptables*-Konfiguration wie folgt aussehen:

```
iptables -t nat -A PREROUTING -i enp0s3 -p tcp --dport 80 -j REDIRECT --to-port 3128
iptables -t nat -A PREROUTING -i enp0s3 -p tcp --dport 8080 -j REDIRECT \
--to-port 3128
iptables -t nat -A POSTROUTING -p tcp --dport 443 -j SNAT --to-source 10.0.0.1
```

Listing 13.47 Weiterleitung von TCP-Port 80 und 8080 an Port 3128 und NAT für Port 443

Aktivieren Sie das `ip_forward`, damit Ihr Proxy-Server das NAT durchführen kann. Dies kann entweder nur zur Laufzeit des Systems über `/proc` geschehen:

```
root@web-proxy:~# echo "1" > /proc/sys/net/ipv4/ip_forward
```

Listing 13.48 Aktivierung: »`ip_forward`«

Oder Sie richten es dauerhaft ein, und zwar über die Datei `/etc/sysctl.conf` bzw. in einer eigenen Datei unterhalb von `/etc/sysctl.d/` mit dem Inhalt aus Listing 13.49:

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Listing 13.49 Dauerhafte Aktivierung mit »`/etc/sysctl.conf`«: »`ip_forward`«

Keine Authentifizierung möglich!

Da der Client nicht weiß, dass er über einen Proxy Inhalte abrufen will, greifen die entsprechenden Mechanismen nicht, sodass keine Authentifizierung erfolgen kann.



13.8 Ab in den Pool – Verzögerung mit `delay_pools`

Wenn Sie den Webverkehr teilweise »künstlich« verzögern oder Benutzern eine Volumenbegrenzung zuteilen wollen, können Sie dies ebenfalls mit dem Squid erreichen. Hier kommen die `delay_pools` ins Spiel. Erstmals wurden `delay_pools` im Squid ab Version 2.x eingeführt. In diesem Abschnitt blicken wir tief in den Eimer.

13.8.1 Funktionsweise – alles im Eimer!

Verzögerungen, Eimer, Datendurchsatz und -anzahl – was soll das Ganze? Da der Web-Proxy auf Applikationsebene arbeitet, kann er nicht, wie etwa ein Router, die Pakete auf TCP-Ebene verlangsamen. Allerdings kann die Verarbeitung der Anfragen verzögert werden. Um Ihnen trotzdem ein Mittel an die Hand zu geben, die Geschwindigkeit des Datenverkehrs zu reglementieren, wurde das komplexe System der `delay_pools` implementiert. Stellen Sie sich den Datendurchfluss durch Ihren Web-Proxy als Flüssigkeit vor. Je nach Konfiguration generiert der Squid eine Anzahl von Eimern. Der Datenverkehr des Web-Proxys wird dann entsprechend der Konfiguration in diese Eimer eingeteilt und an die Benutzer ausgeschüttet.

Dabei wird nicht wirklich der Datenfluss (oder der Datendurchsatz, wie z. B. 127 Kbit/s) reglementiert, sondern die Datenanzahl (259 KB), die in einer Sekunde durch den Proxy fließt. Die Zuteilung zu den jeweiligen Eimern ist auf den ersten Blick etwas kompliziert. Dankenswerterweise wird die Steuerung, welcher Verkehr verzögert wird, über ACLs realisiert, sodass Sie explizit definieren können, wann und wie die Einschränkung erfolgt und was sie bewirkt.



Es gibt allerdings auch Einschränkungen beim Betrieb von *delay_pools*:

- ▶ **Transferrate**
Pools schränken nur die tatsächliche Transferrate ein und lassen dabei Overhead unberücksichtigt (zum Beispiel TCP, ICP, DNS).
- ▶ **Kollektivbestrafung**
Wenige große Transfers von einzelnen Benutzern können einen gemeinsamen Eimer vollständig leeren. Dann können keine weiteren Daten über diesen Eimer abgerufen werden – neue Anfragen werden vollständig abgelehnt.
- ▶ **IPv6**
Mit dem Squid in Version 3.1 können noch keine IPv6-Subnetze verwendet werden.
- ▶ **Eimeranzahl (Squid-Version < 3.1)**
Ältere Squid-Versionen können nur Eimer bis 32 Bit adressieren.

13.8.2 Details – Klassen, Eimer und ACLs richtig wählen

Bevor wir die Details erläutern, zeigen wir Ihnen zunächst eine vollständige *delay_pool*-Konfiguration. In dem Beispiel aus Listing 13.50 werden zwei Pools angelegt, und es wird je einer den Server- und Mitarbeiternetzen zugewiesen:

```
delay_pools 2
delay_class 1 1
delay_class 2 3
delay_parameters 1 65536/1048576
delay_parameters 2 65536/1048576 -1/-1 16384/262144
acl Server src 10.0.0.0/24
acl MA src 192.168.0.0/24
delay_access 1 allow Chef
delay_access 2 allow MA
```

Listing 13.50 »delay_pool«-Konfiguration in der »squid.conf«

delay_pools

Mit dem Parameter `delay_pools N` setzen Sie die Anzahl der Pools fest, die der Squid verwenden soll. Wenn Sie zwei Pools verwenden wollen, sollte die entsprechende Zeile in Ihrer *squid.conf* also `delay_pools 2` lauten.

delay_class

Jeder Pool kann einer unterschiedlichen Klasse angehören. Sie müssen jedem Pool eine Klasse zuweisen. In Listing 13.50 wird dem ersten Pool die Klasse 1 und dem zweiten Pool die Klasse 3 zugewiesen. Wie bereits erwähnt wurde, findet die Einteilung von Pools in Klassen statt. Diese beeinflussen die Anzahl der zur Verfügung gestellten Eimer wie folgt:

- ▶ **class 1**
einfacher Eimer, der für alle Anfragen genutzt wird
- ▶ **class 2**
ein Gesamteimer und je ein Eimer für die Clients eines IPv4-Class-C-Netzes
- ▶ **class 3**
255 Eimer für ein IPv4-Class-B-Netz und je ein Eimer für jeden Client
- ▶ **class 4**
wie Klasse 3, aber zusätzlich ein Eimer je authentifiziertem Benutzer
- ▶ **class 5**
selbst definierte Klasse, die anhand des Rückgabewerts tag des external_acl_type-Parameters aus den http_access-Parametern gebildet wird – je ein Eimer für jeden Wert

delay_parameters

Anschließend folgt die eigentliche Zuweisung der Einschränkungen. Hierfür wird der Parameter delay_parameters verwendet. Folgende Syntax liegt ihm zugrunde:

```
delay_parameters N rate/size [rate/size [rate/size]]
```

Listing 13.51 Syntax »delay_parameters«

Stolperstein: »rate/size«

Die Angabe von rate/size erfolgt in Bytes pro Sekunde und nicht wie gewöhnlich in Bits pro Sekunde. Achten Sie also darauf, die Angaben entsprechend umzurechnen.

Dabei spezifiziert N den entsprechenden Pool und rate/size die Rate und die maximale Größe des Pools. Je nach Klasse müssen mehrere rate/size-Angaben vorgenommen werden. Es werden die Eimertypen aggregate, individual und network unterschieden.

Klasse/Type	aggregate	individual	network	user	target
1	1	–	–	–	–
2	1	256	–	–	–
3	1	65536	256	–	–
4	1	65536	256	zur Laufzeit	–
5	–	–	–	–	zur Laufzeit

Tabelle 13.13 Anzahl der Eimer je Klasse

Da die Klassen 1 bis 3 fest definiert auf Netzwerkebene arbeiten, sind diese an die möglichen Maximalwerte gebunden (siehe Tabelle 13.13). Die Klassen 4 und 5 hingegen werden zur Laufzeit berechnet.

Da nicht alle Klassen jeden Eimertyp unterstützen, unterscheidet sich auch die Anzahl der zu übergebenden Parameter und somit deren Syntax:

- ▶ **Syntax für die Klasse 1**
delay_parameters pool aggregate
- ▶ **Syntax für die Klasse 2**
delay_parameters pool aggregate individual
- ▶ **Syntax für die Klasse 3**
delay_parameters pool aggregate network individual
- ▶ **Syntax für die Klasse 4**
delay_parameters aggregate network individual user
- ▶ **Syntax für die Klasse 5**
delay_parameters pool target

delay_access

Für wen die Verzögerung angewandt werden soll, wird über den Parameter `delay_access` geregelt. Dieser arbeitet äquivalent zu `http_access` (siehe Abschnitt 13.4.9, »Regeln«). Lediglich die Angabe des Pools wird der eigentlichen Regel vorangestellt. Daraus ergibt sich die Syntax aus Listing 13.52:

```
delay_access N allow/deny ACL [ACL]
```

Listing 13.52 Syntax von »delay_access«

Mit `N` wird der Pool spezifiziert, über `allow` und `deny` wird das Einsortieren in diesen Pool vorgenommen, und mit `ACL` wird angegeben, welches Objekt zur Prüfung verwendet wird.

13.9 Familienbetrieb: Sibling, Parent und Co.

Wie so oft im Leben gilt auch für Web-Proxy-Server der Grundsatz: »Besser mehr als weniger«. Der Squid bringt von Haus aus die Möglichkeit mit, Beziehungen zwischen mehreren Installationen herzustellen. Dabei können Sie zum Beispiel an einem Standort lokale Proxy-Server mit eigenen Regeln etablieren, die ihre Inhalte über einen (oder mehrere) zentrale Web-Proxys abrufen. Ebenso ist es möglich, mehrere Web-Proxy-Server in Ihrem Rechenzentrum aufzusetzen und diese so zu konfigurieren, dass Inhalte zuerst in den anderen Installationen gesucht werden.

13.9.1 Grundlagen

Der Squid verwendet zur Bezeichnung der Beziehungen von Proxys untereinander eigene Begriffe. Anhand von Abbildung 13.4 können Sie die Beziehungen gut nachvollziehen.

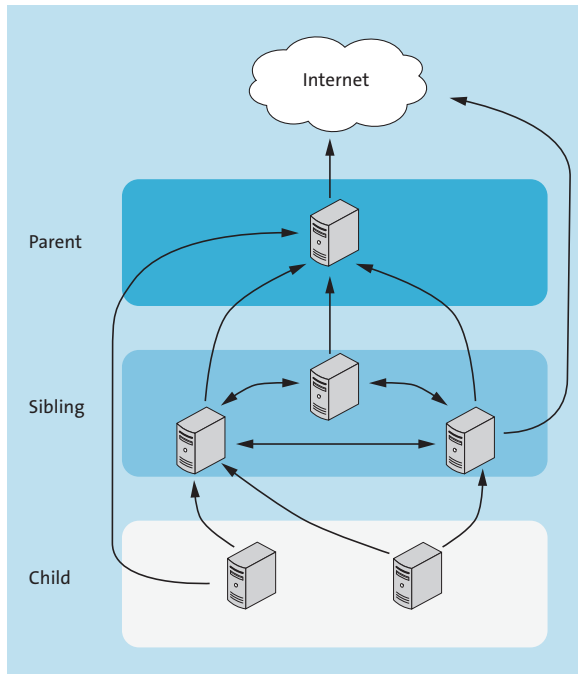


Abbildung 13.4 Squid-Beziehungen

Wie Sie Abbildung 13.4 entnehmen können, sind die Beziehungen nicht hierarchisch. So kann ein Proxy der untersten Ebene auch direkt mit einem der obersten Ebene kommunizieren. Zur Übertragung von Cache-Inhalten von anderen Proxy-Servern wird das ICP (*Internet Cache Protocol*) verwendet, das speziell zur Übertragung von Cache-Inhalten entwickelt wurde. Die Konfiguration erfolgt über den Parameter `cache_peer`.

Der Parameter folgt dabei stets der gleichen Syntax (siehe Listing 13.53):

```
cache_peer <HOSTNAME> <TYPE> <HTTP-PORT> <ICP-PORT> [OPTIONS]
```

Listing 13.53 Syntax: »cache_peer«

Die einzelnen Platzhalter haben dabei nachstehende Bedeutung:

- ▶ <HOSTNAME>
gibt den Hostnamen oder die IP-Adresse des Partners an.
- ▶ <TYPE>
Hier wird die Beziehungsart definiert. Der Squid unterscheidet dabei zwischen:

- parent = übergeordneter oder Eltern-Proxy
 - sibling = gleichgestellter oder Geschwister-Proxy
 - multicast = Rundruf-Proxy gleicher Ebene
- ▶ <HTTP-Port>
setzt den Port, auf dem HTTP-Anfragen gestellt werden.
 - ▶ <ICP-Port>
setzt den Port, auf dem ICP-Anfragen gestellt werden.
 - ▶ [OPTIONS]
Hier können je nach Beziehungstyp weitere Parameter konfiguriert werden.

13.9.2 Eltern definieren

Damit ein Proxy-Server Inhalte nicht direkt, sondern stets von einem übergeordneten Web-Proxy abrufen, müssen Sie lediglich die in Listing 13.54 dargestellte Zeile der *squid.conf* hinzufügen:

```
cache_peer <HOSTNAME> parent 3128 3130 default
```

Listing 13.54 Übergeordnete Proxy-Server konfigurieren

13.9.3 Geschwister definieren

Wenn Sie in Ihrem Rechenzentrum mehrere Proxy-Server betreiben wollen, können Sie zwischen diesen eine Geschwisterbeziehung einrichten. Dafür sind die Zeilen aus Listing 13.55 notwendig – achten Sie darauf, dass diese Zeilen auf allen Proxy-Servern eingerichtet werden müssen und dort jeweils angepasst werden:

```
cache_peer proxy2.example.com sibling 3128 3130 proxy-only  
cache_peer proxy3.example.com sibling 3128 3130 proxy-only  
cache_peer proxy4.example.com sibling 3128 3130 proxy-only
```

Listing 13.55 Gleichgestellten Proxy-Server definieren

Wie Sie Listing 13.55 entnehmen können, sollte dies die Konfiguration des Systems *proxy1.example.com* darstellen, da dort die Geschwisterbeziehung zu *proxy2* bis *proxy4* hergestellt wird.



Besonderheit: »proxy-only«

Beachten Sie, dass in Listing 13.55 als Option *proxy-only* angegeben wurde. Dies veranlasst den lokalen Squid, Inhalte, die von einem gleichgestellten Proxy-Server abgerufen werden,

nicht in den lokalen Cache aufzunehmen. Dies ist extrem wichtig, damit nicht defekte Inhalte in Ihren Cache geraten oder Sie ein leichtes Opfer von Spoofing werden. Darüber hinaus schützt es Sie vor Cache-Schleifen, in denen immer wieder Inhalte aus den Caches geladen und nie neu abgefragt werden!

13.9.4 Load Balancing

Der Parameter `cache_peer` kann mehrfach verwendet werden. Dies kann dazu genutzt werden, ein Load Balancing zu etablieren. Dies kann vor allem dann sinnvoll sein, wenn Sie mehrere Proxy-Server in Ihrem Rechenzentrum vorhalten und in den dezentralen Proxy-Servern in Ihren Außenstellen eine Ausfallsicherheit generieren möchten. Dabei ist der Squid so intelligent zu prüfen, ob der übergeordnete Proxy-Server derzeit verfügbar ist, und diesen gegebenenfalls aus der Liste zu entfernen. Der Squid bietet Ihnen mehrere Mechanismen zum Load Balancing an. Die gängigsten möchten wir Ihnen nun vorstellen:

- ▶ *default*
Dies stellt die Standardkonfiguration dar – kein Load Balancing.
- ▶ *round-robin*
Bei dieser Form wird pro Zugriff ein anderer Proxy-Server verwendet – reihum.
- ▶ *userhash*
Bei dieser Form wird die Auswahl des zu verwendenden Proxy-Servers über eine Hash-Wert-Bestimmung anhand des anfragenden Benutzers getroffen.
- ▶ *sourcehash*
Bei dieser Form wird die Auswahl des zu verwendenden Proxy-Servers über eine Hash-Wert-Bestimmung anhand der IP-Adresse des anfragenden Clients getroffen.

Wir empfehlen Ihnen den Einsatz von *sourcehash* als Load-Balancing-Algorithmus. Dies hat den Vorteil, dass ein Client stets über den gleichen übergeordneten Proxy-Server geleitet wird – was vor allem bei der Fehleranalyse ein Segen ist. Der Algorithmus stellt zwar kein gutes Mittel zur Lastverteilung dar, erhöht aber die Administrationsfreundlichkeit deutlich.

13.9.5 Inhalte eigenständig abrufen: `always_direct`

Eine Besonderheit beim Einsatz von `cache_peer` sind wir Ihnen noch schuldig. Da durch die Konfiguration sämtliche Inhalte über den übergeordneten Proxy-Server geladen werden, ergibt sich teilweise ein unerfreulicher Nebeneffekt. Zum Beispiel würden somit auch lokale Webserver einer Außenstelle immer über den übergeordneten Proxy abgefragt werden. Um dies zu verhindern, wurde die Direktive `always_direct` geschaffen. Die Syntax ist äquivalent zu `http_access` (siehe Listing 13.56):

```
always_direct <TYPE> <ACL>
```

Listing 13.56 Syntax: »always_direct«

Als <TYPE> können `allow` oder `deny` verwendet werden. Über die Option <ACL> wird das Objekt angegeben, auf das diese Regel angewandt wird. Möchten Sie also zum Beispiel verhindern, dass die internen Webserver des Standorts Duisburg (*duisburg.example.com*) über den übergeordneten Proxy abgefragt werden, müsste das Regelwerk wie folgt erweitert werden:

```
acl localServer dstdomain .duisburg.example.com
always_direct allow localServer
```

Listing 13.57 Anpassung des Regelwerks mit »always_direct«

13.10 Cache-Konfiguration

Neben den Objekten und Regeln stellt die korrekte Cache-Konfiguration den ambitionierten Web-Proxy-Administrator vor eine große Herausforderung. Ein falsch dimensionierter Cache kann die Performance Ihres Web-Proxys sehr stark beeinflussen. In diesem Abschnitt wollen wir Ihnen ein paar Grundregeln mit an die Hand geben, sodass Sie der Herausforderung gelassen entgegentreten können.

13.10.1 Cache-Arten: Hauptspeicher und Festplatten

Der Squid unterscheidet zwei Arten von Caches: zum einen den Festplatten-Cache, der durchaus einen Umfang von mehreren Gigabyte haben kann, und zum anderen den flüchtigen Cache im Hauptspeicher.

Generell arbeitet der Squid so, dass neue oder stark frequentierte Objekte im Hauptspeicher-Cache vorgehalten werden, sodass eine schnelle Auslieferung garantiert ist – diese Cache-Objekte werden auch als *In-Transit* bezeichnet. Alle Objekte, die sich gerade nicht *In-Transit* befinden, können daher in den Festplatten-Cache ausgelagert werden.

Die Bezeichnung *In-Transit* ist streng genommen inkorrekt, da im Hauptspeicher-Cache sowohl *In-Transit*-Objekte (Objekte, die sich gerade in der Zustellung befinden) als auch *Hot Objects* (Objekte, die oft abgefragt werden) und *Negative-Cached Objects* (negative Cache-Objekte, zum Beispiel Fehlermeldungen) vorliegen. Letztere werden aber von *In-Transit*-Objekten verdrängt, falls dies notwendig wird.

Bei Zugriffen auf Webseiten wird jeweils geprüft, ob sich das abgefragte Objekt in einem der Caches befindet, und gegebenenfalls, ob dieses noch gültig ist. Anschließend wird das Objekt entweder direkt aus dem Cache ausgeliefert oder neu abgefragt und erneut im Cache abgelegt.

13.10.2 Hauptspeicher-Cache

Die Größe des Hauptspeicher-Caches wird über den Parameter `cache_mem` definiert. Es hat sich als gute Regel etabliert, den `cache_mem` mit der Hälfte des Ihnen zur Verfügung stehenden Hauptspeichers zu belegen. Falls Sie feststellen, dass die übrigen Prozesse auf dem System weniger speicherhungrig sind, können Sie den Wert erhöhen.

»cache_mem« =! Maximum

Die Programmierer des Squid stellen deutlich klar, dass der dort angegebene Wert *nicht* den maximal vom Squid genutzten Hauptspeicher darstellt! Zum einen, da der Wert nur den Cache im Hauptspeicher angibt und der Prozess durchaus auch andere Daten im Hauptspeicher vorhalten muss, und zum anderen, da der angegebene Wert auch kurzzeitig überschritten werden kann. Planen Sie also einen entsprechenden Puffer ein, denn nichts verlangsamt Ihr System mehr als die Auslagerung des Hauptspeichers auf die Festplatte (*Swapping*).



13

Für einen Server mit 8 GB Hauptspeicher sollte der Parameter dann wie folgt gesetzt sein:

```
cache_mem 4096 MB
```

Listing 13.58 Größe des Hauptspeicher-Caches definieren

Dies ist aber nicht der Weisheit letzter Schluss, sondern nur ein möglicher Startwert. Wenn Sie einen (oder mehrere) Festplatten-Caches einsetzen wollen, richtet sich die maximale Größe des Hauptspeicher-Caches nach den eingerichteten Festplatten-Caches.

Im Unterabschnitt »Größenberechnung« des Abschnitts 13.10.3, »Festplatten-Cache«, gehen wir näher auf die korrekte Berechnung ein.



13.10.3 Festplatten-Cache

Der Festplatten-Cache stellt den Löwenanteil des Caches dar. In ihm werden alle Objekte abgelegt, die über einen entsprechend langen Gültigkeitszeitraum verfügen. Trotz der stetig steigenden Bandbreiten ist es sinnvoll, einen großen Web-Proxy-Cache vorzuhalten, da nichts schneller ist als die lokale Auslieferung.

Der Festplatten-Cache wird über den Parameter `cache_dir` eingerichtet. Im Squid ist es möglich, mehr als einen Festplatten-Cache anzulegen: So können Sie zum Beispiel auf einem System mit mehreren Festplatten pro Festplatte einen eigenen Cache anlegen. Die Syntax des Parameters `cache_dir` sieht für die Methode `aufs` wie folgt aus:

```
cache_dir <METHOD> <DIRECTORY> <SIZE> <L1> <L2>
```

Listing 13.59 Syntax: »cache_dir« für die Methode »aufs«

Dabei haben die einzelnen Platzhalter folgende Bedeutungen:

▶ <METHOD>

gibt die Speichermethode des Caches an. Der Squid versteht diese Methoden:

- ufs
- aufs
- diskd
- rock

Je nach Verwendungszweck ist eine dieser Methoden zu verwenden. Wir empfehlen den Einsatz von `aufs` für einen generellen Cache – jede Methode hat ihre eigene Syntax. In Listing 13.59 wurde daher die Syntax für die Methode `aufs` dargestellt.

▶ <DIRECTORY>

gibt das Verzeichnis an, in dem der Squid den Cache verwalten soll.

▶ <SIZE>

gibt die Gesamtgröße des Caches an.

▶ <L1>

Anzahl der Verzeichnisse der ersten Ebene

▶ <L2>

Anzahl der Verzeichnisse der zweiten Ebene – Anzahl von Verzeichnissen unterhalb von jedem L1-Verzeichnis



Es empfiehlt sich, für einen Festplatten-Cache stets eine eigene Festplatte zu verwenden! Dies hat den großen Vorteil, dass sich die Zugriffszeiten auf die Inhalte nicht durch das System oder andere Dienste verringern. Am besten ist selbstverständlich ein Cache auf einem Solid-State-Drive (SSD), das über ein Hardware-RAID auf mehrere Festplatten verteilt ist.



Besonderheiten der Formatierung: »Inodes«

Bedenken Sie, dass in Ihren Festplatten-Cache viele kleine Dateien geschrieben werden. Daher ist es notwendig, dass das Filesystem eine entsprechend große Anzahl von *Inodes* zur Verfügung stellt. Für einen Festplatten-Cache auf dem Device `sdb1` mit einem *Ext4*-Filesystem müsste die Formatierung dann mit `mkfs.ext4 -N 53772288 /dev/sdb1` erfolgen. Durch den Parameter `-N` wird die Anzahl der *Inodes* angegeben.

Größenberechnung

Damit Ihr Cache Ihnen nicht die Festplatte zu 100 % auslastet oder aufgrund des zu geringen Hauptspeichers mehr Verwaltungsaufwand als Nutzen bringt, muss er korrekt eingerichtet werden. Zusätzlich besteht die Gefahr, dass Ihr Web-Proxy nicht über genügend Hauptspei-

cher verfügt. Die korrekten Berechnungen des eingesetzten Hauptspeicher- und Festplatten-Caches bedingen sich gegenseitig. Beginnen wir mit der Berechnung des Festplatten-Caches.

Ein Festplatten-Cache mit der Methode *aufs* erwartet die Optionen zu Größe und Anzahl der Verzeichnisse der ersten Ebene und die Anzahl der Verzeichnisse der zweiten Ebene. Um diese Werte korrekt zu bestimmen, gibt es eine Faustformel:

$$\langle \text{Cache-Größe in KB} \rangle / (\langle L1 \rangle \times \langle L2 \rangle \times \langle \text{Durchschnittliche Objektgröße in KB} \rangle) \sim 200$$

Listing 13.60 Faustformel: Festplatten-Cache

In Pseudocode bedeutet dies: »Die Größe des Caches in Kilobyte, geteilt durch die Multiplikation der Anzahl der Verzeichnisse der ersten Ebene, der Anzahl der Verzeichnisse der zweiten Ebene und der durchschnittlichen Objektgröße in Kilobyte, sollte in etwa 200 ergeben.«

Zugegeben: Dies klingt komplex, ist aber in der Anwendung einfacher als gedacht. Der Hintergedanke dieser Formel besteht darin, dass pro Verzeichnis der zweiten Ebene nicht mehr als 200 Objekte gespeichert werden sollen, da dies sonst ineffektiv werden würde. Ohne vorherige Kenntnis der durchschnittlichen Objektgröße können wir den Wert 16 KB verwenden.

Anhand dieser Formel aus Listing 13.60 kann die maximale Größe des Festplatten-Caches ermittelt werden. Dabei ist aber darauf zu achten, dass Sie nicht einfach die maximale Größe Ihrer Festplatte verwenden können: Auch wenn Sie eine 1 TB große Festplatte extra für den Cache abgestellt haben, ist es nicht automatisch sinnvoll, auch einen 1 TB großen Cache anzulegen. Wie bereits angedeutet wurde, bedingen sich die Größe des Festplatten-Caches und die des Hauptspeicher-Caches. Da ein Festplatten-Cache auch verwaltet werden möchte, gilt der Grundsatz »10 MB Hauptspeicher pro 1 GB an Festplatten-Cache«. Zusätzlich benötigt der Squid noch eine Hauptspeicherreserve, damit der Prozess selbst auch noch »Luft zum Atmen« hat.

Bei einem Server mit 8 GB Hauptspeicher, von denen 4 GB dem Hauptspeicher-Cache (*cache_mem*) zugewiesen wurden, stehen noch ca. 4 GB zur Verfügung. Die Hälfte davon sollte für das Betriebssystem und andere Dienste frei gelassen werden. Daher stehen für den Festplatten-Cache noch ca. 2 GB zur Verfügung. Da pro Gigabyte an Festplatten-Cache ca. 10 MB an Hauptspeicher benötigt werden, ergibt sich daraus eine maximale Größe des Festplatten-Caches von 204,8 GB (2048 MB / 10 MB). Um einen entsprechenden Puffer zu haben, sollten Sie den Festplatten-Cache also nicht größer als 200 GB gestalten.

In der *squid.conf* könnte diese Konfiguration dann wie folgt aussehen, wobei sich die Werte für *L1* und *L2* aus der Formel $204800 \times 1024 / (64 \times 1024 \times 16)$ ergeben:

```
cache_mem 4096 MB
cache_dir aufs /var/spool/squid 204800 64 16
```

Listing 13.61 Cache-Konfiguration mit 8 GB Hauptspeicher

Anlegen

Damit der Squid den Festplatten-Cache verwendet, muss dieser zunächst angelegt werden. Dafür müssen Sie den Dienst stoppen und mit dem Befehl `squid -z` die Verzeichnisse vom Squid erzeugen lassen.



Achten Sie darauf, dass der Benutzer, unter dem der Squid läuft – standardmäßig `proxy` oder `squid` –, auch das Recht besitzt, auf das Verzeichnis zuzugreifen, in dem sich der Cache befinden soll.

Anschließend können Sie den Dienst neu starten. Der Cache wird dann sukzessive vom Squid gefüllt werden.

13.10.4 Tuning

Wie bereits erörtert wurde, ist nichts schneller als die lokale Zustellung. In der Standard-einstellung lässt der Squid keine Objekte im Hauptspeicher, die größer sind als 512 KB, und keine Objekte im Festplatten-Cache, die größer sind als 4096 KB. Selbstverständlich können Sie diese Werte an Ihre Bedürfnisse anpassen.



Beachten Sie aber, dass dies Auswirkungen auf die mögliche Nutzung des Caches hat! Wenn Sie lieber viele kleine Dateien schnell ausliefern und somit eine größere Trefferanzahl (auch *Hit Ratio* genannt) erlangen wollen, sollten Sie die Werte kleiner lassen. Möchten Sie hingegen Bandbreite sparen, können Sie diese Werte auch höher konfigurieren.

Die maximale Größe von Objekten im Hauptspeicher wird über den Parameter `maximum_object_size_in_memory` gesteuert. Die maximale Größe der Objekte im Festplatten-Cache wird über den Parameter `maximum_object_size` gesteuert.

Kapitel 14

Kerberos

Wenn Sie in Ihrem Netzwerk verschiedene Dienste anbieten, für die eine Authentifizierung benötigt wird, ist es sinnvoll, über ein Single Sign-on mithilfe von Kerberos nachzudenken. Ihre Anwender werden es Ihnen danken, wenn sie sich nur noch einmal anmelden müssen und dann alle Dienste im Netz nutzen können.

Kerberos wurde 1978 am *Massachusetts Institute of Technology* (MIT) entwickelt und ist ein Authentifizierungsdienst für Netzwerkdienste. Die aktuelle Version von Kerberos ist die Version 5. Die wohl gebräuchlichste Implementierung von Kerberos ist die vom *Massachusetts Institute of Technology*. Weil diese Version aber unter strengen Exportregeln der USA stand, wurden verschiedene andere Implementierungen außerhalb der USA entwickelt. Die meistverwendete ist hier die *Heimdal*-Implementierung aus Schweden. Seit dem Jahr 2000 gilt die strenge Reglementierung für den MIT-Kerberos nicht mehr und er kann auch außerhalb der USA verwendet werden. In diesem Kapitel werden wir uns ausschließlich mit der MIT-Implementierung beschäftigen, da diese in allen Distributionen vorhanden ist.

Neben diesen beiden gibt es noch einige andere weniger gebräuchliche Implementierungen. Auch Microsoft verwendet im *Active Directory* eine Art der Kerberos-Authentifizierung. Aber diese ist nicht kompatibel mit den unter Linux verwendeten Implementierungen. Im *Active Directory* wurde die MIT-Kerberos-Version um einige Funktionen erweitert.

Die Verwendung von Kerberos bietet Ihnen eine einheitliche und sichere Authentifizierungsmethode. Bei einer Authentifizierung in einem Netzwerk, das mit Kerberos arbeitet, wird die Authentifizierung von einer *vertrauenswürdigen dritten Person (Trusted Third Party)* übernommen. Bei dieser handelt es sich dann um den *Kerberos-Dienst*.

Der *Kerberos-Server* wird nur für die Authentifizierung sorgen, er kann jedoch nicht zur Autorisierung oder zum Verwalten der Benutzerkonten eingesetzt werden. Für die Aufgaben der Kontenverwaltung können Sie einen LDAP-Server verwenden. Mehr dazu finden Sie in Kapitel 17, »LDAP«. Am Ende dieses Kapitels finden Sie aber die Zusammenführung von Kerberos und LDAP. Mit der Hilfe von Kerberos können Sie in Ihrem Netzwerk ein *Single Sign-on (SSO)* realisieren: Jeder Benutzer gibt sein Passwort nur einmal ein und erhält dann ein *Ticket*. Mit diesem Ticket kann er sich dann automatisch bei allen Diensten im Netzwerk anmelden, die Kerberos unterstützen.

14.1 Begriffe im Zusammenhang mit Kerberos

Beim Einsatz von Kerberos werden verschiedene Begriffe verwendet. Deshalb erhalten Sie hier eine Übersicht über die Begriffe und deren Bedeutung:

► **Realm**

Bei dem *Realm* handelt es sich um den administrativen Bereich für den Kerberos-Server. Dieser Bereich wird oft auch *Authentifizierungszone* genannt. Hierfür wird häufig der Name der DNS-Domäne in Großbuchstaben verwendet. Achten Sie bei der Verwendung des Realms immer darauf, dass Sie diesen auch großschreiben.

► **Principal**

Der *Principal* wird von Kerberos für die Authentifizierung benötigt. Der Principal setzt sich aus einer oder mehreren Komponenten und dem Realm zusammen. Ein Principal für einen Benutzer hat nur eine Komponente, die mit dem @-Zeichen an den Realm gebunden wird (zum Beispiel stefan@example.net). Für die Authentifizierung eines Service- oder Hosttickets werden dagegen zwei Komponenten verwendet (zum Beispiel host/adminbuch.example.net@EXAMPLE.NET).

► **Ticket**

Das *Ticket* wird für die Anmeldung an einem Host oder für einen Dienst benötigt. Das *Ticket* kann auch als virtueller Fahrschein betrachtet werden, ohne den der Dienst nicht benutzt werden kann.

► **Ticket Granting Ticket (TGT)**

Das TGT erhält der Client vom *Ticket Granting Server (TGS)*. Mit diesem TGT kann der Client, ohne weitere Authentifizierungen durchführen zu müssen, die Tickets für weitere Dienste erhalten.

► **Authentication Service (AS)**

Der AS beantwortet die Anfragen der Clients und erzeugt die zufälligen Sitzungsschlüssel sowie die Kerberos-Tickets.

► **Ticket Granting Server (TGS)**

Der TGS vergibt das *Ticket Granting Ticket (TGT)* an die Clients. Mithilfe dieses Tickets kann dann der Client Tickets für andere Dienste beziehen.

► **Key Distribution Center (KDC)**

Hierbei handelt es sich um eine Datenbank, in der alle Principals mit ihrem Passwort abgelegt sind. Der TGS ist ein Teil des KDC.

14.2 Die Funktionsweise von Kerberos

Bevor ein Dienst oder ein Host Kerberos für die Authentifizierung von Benutzern über Tickets nutzen kann, muss der entsprechende Dienst *kerberized* werden.

Denn nur dann kann der Dienst das Ticket des Anwenders gegen den Kerberos-Server überprüfen. Wenn sich ein Anwender am System anmeldet, erhält er vom AS ein TGT. Will nun der Anwender einen Dienst nutzen oder auf einen anderen Host zugreifen, der *kerberized* ist, holt sich der Client mit dem TGT ein Ticket für den entsprechenden Host oder Dienst. Der Client schickt dann das Ticket an den Host oder Dienst. Dieser kann dann das Ticket gegen den Kerberos-Server prüfen und daraufhin den Zugriff zulassen. Dieser Vorgang findet für den Anwender absolut transparent statt.

Einstufiges Kerberos-Verfahren

Beim einstufigen Kerberos-Verfahren werden alle Authentifizierungen nur vom AS gegen die KDC-Datenbank durchgeführt. Jeder Zugriff von einem Client auf einen Dienst erfordert immer das Passwort des Benutzers. Ein SSO ist mit diesem Verfahren nicht realisierbar, da kein TGT an den Client vergeben wird, mit dem er sich automatisch authentifizieren könnte. Dieses Verfahren wird heute kaum noch verwendet und soll deshalb hier nicht weiter besprochen werden.

Zweistufiges Kerberos-Verfahren

Beim zweistufigen Kerberos-Verfahren kommt der TGS ins Spiel. Jeder Client, der sich mit seinem in der KDC-Datenbank eingetragenen Principal und dessen Passwort angemeldet hat, erhält ein TGT, das nur eine begrenzte Gültigkeitsdauer hat (meist acht bis zehn Stunden). Dieses Ticket wird im Ticket-Cache des Benutzers abgelegt. Immer wenn der Anwender jetzt auf einen neuen Dienst oder Host zugreifen will, wird der Kerberos-Client im Hintergrund mithilfe des TGT ein Ticket für den Zugriff vom TGS anfordern und damit die Authentifizierung für den Anwender transparent durchführen. Damit ist ein echtes SSO möglich.

Bevor wir jetzt zur Installation und Konfiguration des Kerberos-Servers kommen, noch ein wichtiger Punkt: Kerberos ist sehr zeitkritisch. Bei der Überprüfung der Tickets wird auch immer die Zeit der Erstellung überprüft. Sie sollten aus diesem Grund unbedingt einen Zeitserver in Ihrem Netzwerk haben, und alle Hosts in Ihrem Netzwerk sollten sich die Systemzeit von diesem Zeitserver holen. Auch sollte ein funktionsfähiger DNS-Server in Ihrem Netzwerk laufen, der alle Server- und Clientadressen korrekt auflösen kann. Wichtig ist hier, dass sowohl die Vorwärtsauflösung als auch die Rückwärtsauflösung funktioniert.



14.3 Installation und Konfiguration des Kerberos-Servers

Bevor Sie mit der Konfiguration beginnen, installieren Sie die benötigten Pakete. Für Debian und Ubuntu installieren Sie die Pakete wie gewohnt mit `apt`, wie in Listing 14.1 zu sehen ist:

```
root@kerberos01:~# apt install krb5-admin-server krb5-kdc krb5-user
```

Listing 14.1 Installation der benötigten Pakete bei Debian

Das Paket *krb5-admin-server* dürfen Sie nur auf einem Kerberos-Server installieren, da es in einem Realm nur einen Admin-Server geben kann.



Wenn Sie Ihren Kerberos in den Verzeichnisdienst LDAP einbinden wollen, benötigen Sie noch das Paket *krb5-kdc-ldap*. Mehr dazu, wie Sie Kerberos in LDAP integrieren können, finden Sie in Abschnitt 14.11, »Kerberos in LDAP einbinden«.



Fehler beim Start des KDC

Bei älteren Debian- und Ubuntu-Versionen wird der KDC sofort nach seiner Installation gestartet, das schlägt aber fehl. Diese Fehlermeldung können Sie an dieser Stelle ignorieren, da der Fehler zu erwarten war, denn der KDC ist zu diesem Zeitpunkt noch nicht vollständig konfiguriert. Somit kann der Start nur fehlschlagen. Bei den hier im Buch vorgestellten Versionen ist der KDC-Dienst nicht sofort aktiv, er muss erst über den *systemd* aktiviert werden.

14.3.1 Starten und Stoppen der Dienste

In den verschiedenen Distributionen werden die beiden Dienste *kdc* und *kadmin* über verschiedene *systemd*-Skripte gestartet und gestoppt. Tabelle 14.1 zeigt die *systemd*-Services.

Distribution	Dienst	systemd-Service
Debian/Ubuntu	KDC	krb5-kdc.service
Debian/Ubuntu	kadmin	krb5-admin-server.service
openSUSE/CentOS	KDC	krb5kdc.service
openSUSE/CentOS	kadmin	kadmin.service

Tabelle 14.1 *systemd*-Services für Kerberos

Bei Debian und Ubuntu aktivieren Sie den KDC und der Adminserver über den *systemd*, auch bei openSUSE und CentOS die Dienste nicht automatisch aktiviert. Bei openSUSE aktivieren Sie den benötigten Dienste über den *YaST*. Bei CentOS nutzen Sie ebenfalls den *systemd*.

Während der Installation werden die folgenden Parameter abgefragt, über die Sie sich im Vorfeld Gedanken machen sollten:

► DRealm

Mit dem Realm legen Sie den Administrationsbereich für Kerberos fest. Hier tragen Sie den FQDN des Servers ein, der für den Realm verantwortlich ist. Als Realm wird meist der Name der DNS-Domäne verwendet. Das hat den Vorteil, dass Sie keine verschiedenen Namensräume verwalten müssen.

► Administrationsserver

Auf dem Administrationsserver werden die Principals verwaltet. Diesen Server darf es für jeden Realm nur einmal geben. Auch hier müssen Sie den FQDN des Servers angeben.

► KDC-Server

Das ist der Server, auf dem sich die Datenbank mit den Principals befindet. Der Server mit der Datenbank kann ein anderer Server sein als der Server, von dem aus die Datenbank verwaltet wird. Aus diesem Grund wird während der Einrichtung nach dem Admin-Server und dem KDC-Server gefragt. Für das folgende Beispiel sollen beide Dienste auf derselben Maschine eingerichtet werden.

Bei openSUSE installieren Sie die Pakete `krb5-server` und `krb5-client`. Für die Verwendung von LDAP in Verbindung mit Kerberos benötigen Sie zusätzlich das Paket `krb5-plugin-kdb-ldap`. Bei CentOS benötigen Sie die Pakete `krb5-server` und `krb5-workstation` und für die LDAP-Anbindung das Paket `krb5-server-ldap`

Denken Sie beim Einsatz von openSUSE daran, die Dienste `kadmind` und `krb5kdc` im *Run-level-Editor* des YaST zu aktivieren.



Nach der Installation der Pakete können Sie mit der Konfiguration des Servers beginnen. Bei der Installation haben Sie bereits den Realm für Ihren Kerberos festgelegt. Hier im Buch verwenden wir den Realm `EXAMPLE.NET`. Für die Konfiguration des Kerberos-Servers werden die Datei `kdc.conf` und die Datei `/etc/krb5.conf` verwendet. Die Datei `kdc.conf` finden Sie bei Debian und Ubuntu im Verzeichnis `/etc/krb5kdc`, bei openSUSE im Verzeichnis `/var/lib/kerberos/krb5kdc` und bei CentOS im Verzeichnis `/var/kerberos/krb5kdc/`.

14.3.2 Konfiguration der Datei »/etc/krb5.conf«

In dieser Datei werden die Informationen für die Kerberos-Librarys abgelegt sowie die Beschreibung des Realms. Auch der KDC, der für den Realm verantwortlich ist, wird hier eingetragen. Während der Installation der Pakete wird bereits eine Datei `krb5.conf` angelegt. In dieser Datei befinden sich aber sehr viele nicht benötigte Einträge. Aus diesem Grund sollten Sie die Datei immer an Ihre Umgebung anpassen. In Listing 14.2 sehen Sie den Aufbau der Datei für die Beispielumgebung hier im Buch. Löschen oder verschieben Sie die Datei aus der Installation und erstellen Sie eine neue:

```
[libdefaults]
    default_realm = EXAMPLE.NET

[realms]
    EXAMPLE.NET = {
        kdc = kerberos01.example.net
        admin_server = kerberos01.example.net
    }
```

```
[domain_realm]
    .example.net = EXAMPLE.NET

[logging]
    kdc = FILE:/var/log/krb5/kdc.log
    admin_server = FILE:/var/log/krb5/kadmind.log
    default = SYSLOG:NOTICE:DAEMON
```

Listing 14.2 Inhalt der Datei »/etc/krb5.conf«



Schreiben ins Log

Damit die Kerberos-Dienste überhaupt ins Log schreiben können, ist es notwendig, dass Sie in den systemd-Skripten `/usr/lib/systemd/system/krb5-kdc.service` und `usr/lib/systemd/system/krb5-admin-server.service` die Variable `ReadWriteDirectories` um das Verzeichnis `/var/log` und `/var/log/krb5` erweitern. Ein Setzen von Berechtigungen an den Verzeichnissen reicht nicht aus. Nach der Änderung der Dateien laden Sie die Änderungen in den systemd mit dem Kommando `systemctl daemon-reload`. Anschließend starten Sie die beiden Dienste neu.

Die einzelnen Abschnitte und Parameter haben folgende Bedeutungen:

- ▶ `[libdefaults]`
In diesem Abschnitt befinden sich die Standardeinstellungen für die Kerberos-Librarys.
- ▶ `default_realm = EXAMPLE.NET`
Dieser Parameter gibt den Standard-Realm für die Client-Server-Kommunikation an.
- ▶ `[realms]`
In diesem Abschnitt werden alle *Realms* verwaltet, die ein Client kennt, und es wird definiert, auf welchen Hosts sich die entsprechenden Server befinden.
- ▶ `kdc = kerberos01.example.net`
Das ist der FQDN des KDC.
- ▶ `admin_server = kerberos01.example.net`
Da der Admin-Server und der KDC auf verschiedenen Maschinen laufen können, muss hier der FQDN des Admin-Servers angegeben werden.
- ▶ `[domain_realm]`
Hier wird die Verbindung des Kerberos-Realms und des Domainnamens hergestellt. Dieser Eintrag wird von Programmen benötigt, die feststellen müssen, zu welchem Realm ein Host gehört. Hier wird der FQDN der Domain verwendet.
- ▶ `.example.com = EXAMPLE.NET`
Da die Domain hier im Buch genauso heißt wie der Realm, steht auf beiden Seiten der Zuweisung derselbe Name. Wichtig ist, dass der Realm komplett großgeschrieben wird.

- ▶ [logging]
In diesem Abschnitt wird das Logging für den KDC und den Admin-Server festgelegt.
- ▶ kdc = FILE:/var/log/krb5/kdc.log
Dies ist die Log-Datei für den KDC. Das Verzeichnis */var/log/krb5* existiert nur bei openSUSE; auf einem Debian- oder Ubuntu-System müssen Sie das Verzeichnis noch anlegen.
- ▶ admin_server = FILE:/var/log/krb5/kadmind.log
Dies ist die Log-Datei für den Admin-Server. Achten Sie auch hier auf den Pfad, um sicherzugehen, dass das Verzeichnis */var/log/krb5* existiert. Bei Debian und Ubuntu legen Sie das Verzeichnis von Hand an.
- ▶ default = SYSLOG:NOTICE:DAEMON
An dieser Stelle können Sie das Verhalten des *Syslogd* bestimmen, wie beispielsweise hier das Log-Level NOTICE.

Hier noch einmal der Hinweis: Achten Sie darauf, dass die Namen des KDC und des Admin-Servers über DNS auflösbar sind.



14.3.3 Konfiguration der Datei »kdc.conf«

Während der Installation der Pakete wurde der Realm bereits abgefragt. In der Datei *kdc.conf* finden Sie also schon Einträge für den von Ihnen angegebenen Realm.

Die Datei auf einem Debian- oder Ubuntu-System sieht so aus wie in Listing 14.3. Lediglich der Abschnitt zum Logging wurde hier ergänzt:

```
[kdcdefaults]
    kdc_ports = 750,88

[realms]
    EXAMPLE.NET = {
        database_name = /var/lib/krb5kdc/principal
        admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
        acl_file = /etc/krb5kdc/kadm5.acl
        key_stash_file = /etc/krb5kdc/stash
        kdc_ports = 750,88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = aes256-cts-hmac-sha384-192
        supported_encetypes = aes256-cts-hmac-sha384-192:normal \
                            aes128-cts-hmac-sha256-128:normal
        default_principal_flags = +preauth
    }
```

```
[logging]
    kdc = FILE:/var/log/krb5/krb5kdc.log
    admin_server = FILE:/var/log/krb5/kadmind.log
```

Listing 14.3 Inhalt der Datei »/etc/krb5kdc/kdc.conf« auf einem Debian-System

Unter openSUSE hat die Datei den Inhalt aus Listing 14.4:

```
[kdcdefaults]
    kdc_ports = 750,88

[realms]
#     EXAMPLE.COM = {
#         database_name = /var/lib/kerberos/krb5kdc/principal
#         admin_keytab = FILE:/var/lib/kerberos/krb5kdc/kadm5.keytab
[...]
#         kdc_ports = 750,88
#         max_life = 10h 0m 0s
#         max_renewable_life = 7d 0h 0m 0s
#     }

[logging]
    kdc = FILE:/var/log/krb5/krb5kdc.log
    admin_server = FILE:/var/log/krb5/kadmind.log
```

Listing 14.4 Inhalt der Datei »kdc.conf« auf einem openSUSE-System

Leider ist die Datei so völlig unbrauchbar. Das ändern die Einträge aus Listing 14.5:

```
[kdcdefaults]
    kdc_ports = 750,88

[realms]
    EXAMPLE.NET = {
        database_name = /var/lib/kerberos/krb5kdc/principal
        admin_keytab = FILE:/var/lib/kerberos/krb5kdc/kadm5.keytab
        acl_file = /var/lib/kerberos/krb5kdc/kadm5.acl
        key_stash_file = /var/kerberos/krb5kdc/.k5.EXAMPLE.NET
        kdc_ports = 750,88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = aes256-cts-hmac-sha384-192
        supported_encetypes = aes256-cts-hmac-sha384-192:normal \
                            aes128-cts-hmac-sha256-128:normal
        default_principal_flags = +preauth
    }
```

```
[logging]
kdc = FILE:/var/log/krb5/krb5kdc.log
admin_server = FILE:/var/log/krb5/kadmind.log
```

Listing 14.5 Nutzbare Konfiguration

Fehlt nur noch der Inhalt der Datei *kdc.conf* auf einem CentOS-System. Den Inhalt sehen Sie in Listing 14.6:

```
[kdcdefaults]
kdc_ports = 88
kdc_tcp_ports = 88
spake_preauth_kdc_challenge = edwards25519

[realms]
EXAMPLE.COM = {
    acl_file = /var/kerberos/krb5kdc/kadm5.acl
    key_stash_file = /var/lib/kerberos/krb5kdc/.k5.EXAMPLE.NET
    dict_file = /usr/share/dict/words
    admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
    supported_encetypes = aes256-cts-hmac-sha384-192:normal \
                          aes128-cts-hmac-sha256-128:normal
}
```

```
[logging]
kdc = FILE:/var/log/krb5/krb5kdc.log
admin_server = FILE:/var/log/krb5/kadmind.log
```

Listing 14.6 Inhalt der Datei »kdc.conf« auf einem CentOS-System

Diese Datei wurde von uns um die Zeile für das *stash*-File und das Logging erweitert. Legen Sie das Verzeichnis für das Logging an, bevor Sie den KDC neu starten. Alle anderen Werte können Sie so übernehmen.

Achten Sie auf die unterschiedlichen Pfade

Die Pfade unter openSUSE und CentOS unterscheiden sich von denen unter Debian und Ubuntu. Setzen Sie auf jeden Fall die richtigen Pfade für Ihre Distribution!



Die Parameter der Datei haben folgende Bedeutungen:

- ▶ [kdcdefaults]
 - In diesem Abschnitt werden alle Parameter verwaltet, die unabhängig von allen verwalteten *Realms* auf dem Server vom KDC benötigt werden.

- ▶ `kdc_ports = 750,88`

Das sind die Ports, auf denen der KDC Anfragen entgegennimmt. Hierbei handelt es sich um UDP-Ports. Diese Ports können Sie mit dem Kommando `netstat -u|pn` testen. In der daraus resultierenden Auflistung sehen Sie die beiden UDP-Ports 750 und 88, die vom KDC verwendet werden. Diese beiden Ports sind die Standardports für den KDC.
- ▶ `[realms]`

In diesem Bereich werden die Parameter für alle *Realms* verwaltet, für die der KDC verantwortlich ist.
- ▶ `EXAMPLE.NET =`

Das ist der Realm, für den dieser KDC verantwortlich ist. In geschweiften Klammern werden die für diesen Realm relevanten Parameter eingetragen.
- ▶ `database_name = /var/lib/krb5kdc/principal`

Das sind die Namen und der Pfad zur Kerberos-Datenbank.
- ▶ `admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab`

Hierbei handelt es sich um die Datei, die der Admin der Datenbank benutzt, um sich am Kerberos zu authentifizieren.
- ▶ `acl_file = /etc/krb5kdc/kadm5.acl`

Das ist die Datei, in der die Zugriffsregeln auf die Datenbank festgelegt werden.
- ▶ `key_stash_file = /etc/krb5kdc/stash`

In dieser Datei befindet sich das Master-Passwort, das Sie während der Initialisierung der Datenbank festgelegt haben. Bei openSUSE wird hier der Name `.k5.EXAMPLE.NET` verwendet.
- ▶ `kdc_ports = 750,88`

Das sind die Ports, über die dieser Realm erreicht werden kann.
- ▶ `max_life = 10h 0m 0s`

Hier geben Sie an, wie lange ein Ticket gültig ist. 10 Stunden ist der Standardwert.
- ▶ `max_renewable_life = 7d 0h 0m 0s`

Bei diesem Wert geben Sie an, wie lange ein Ticket maximal verlängert werden kann, bevor es neu angefordert werden muss.
- ▶ `master_key_type = aes256-cts-hmac-sha384-192`

Dieser Parameter legt den Typ der Verschlüsselung für das Master-Passwort fest. Dieser Parameter ist bei openSUSE und CentOS nicht gesetzt. Wollen Sie eine spezielle Verschlüsselung für den Master-Key, müssen Sie den Parameter nachtragen.
- ▶ `supported_encetypes = aes256-cts-hmac-sha384-192:normal aes128-cts-hmac-sha256-128:normal`

Hier können Sie festlegen, welche Verschlüsselungen der Kerberos-Server unterstützen soll. Auch dieser Parameter ist bei openSUSE und CentOS nicht in der Standardkonfiguration vorhanden und muss von Ihnen zusätzlich eingefügt werden.

- ▶ `default_principal_flags = +preauth`

Dieser zusätzliche Parameter, der beim Starten des Kerberos-Servers übergeben wird, sorgt dafür, dass eine Pre-Authentication stattfinden muss. Das erhöht die Sicherheit gegen Offline-Passwortangriffe. Wenn Sie noch Kerberos-V4-Clients im Netz haben, dürfen Sie diesen Parameter nicht setzen. Wie bei den beiden Parametern zuvor ist auch dieser Parameter bei openSUSE und CentOS nicht in der Standardkonfiguration vorhanden und muss von Ihnen gesetzt werden, wenn Sie diese Funktion nutzen wollen.

14.4 Initialisierung und Testen des Kerberos-Servers

Nachdem Sie die beiden Konfigurationsdateien erstellt haben, initialisieren Sie nun die Datenbank für den Kerberos-Server. Während der Initialisierung legen Sie ein Master-Passwort fest. Jede Änderung an der Datei `kdc.conf` erfordert einen Neustart des KDCs.

Wichtiger Hinweis zum Master-Passwort

Mit dem Master-Passwort kann die gesamte Kerberos-Datenbank entschlüsselt werden. Sie sollten an dieser Stelle ein sehr sicheres Passwort wählen und das Passwort nicht weitergeben. Das Passwort kann nicht für die Authentifizierung genutzt werden, sondern nur, um die Datenbank zu entschlüsseln. Verlieren Sie das Passwort, dann ist es nicht mehr möglich, die Datenbank komplett zu entschlüsseln. Eine spätere Migration auf einen anderen Server ist ohne das Master-Passwort auch nicht mehr möglich. Legen Sie deshalb das Passwort unbedingt an einer sicheren Stelle ab.



Jetzt können Sie die Datenbank mit dem Kommando `kdb5_util -s create` erzeugen. Bei der Initialisierung werden Sie auch nach dem Master-Passwort gefragt. Der Parameter `-s` erzeugt das `stash`-File, in dem sich das verschlüsselte Passwort befindet. Den Vorgang der Initialisierung sehen Sie in Listing 14.7:

```
root@kerberos01:/etc/krb5kdc# kdb5_util create -s
Zufällige Daten werden geladen.
Datenbank »/var/lib/krb5kdc/principal« für Realm »EXAMPLE.NET« wird initialisiert,
Hauptschlüsselname »K/M@EXAMPLE.NET«
Sie werden nach dem Master-Passwort der Datenbank gefragt.
Es ist wichtig, dass Sie dieses Passwort NICHT VERGESSEN.
Enter KDC database master key:
Re-enter KDC database master key to verify:
```

Listing 14.7 Initialisierung der Kerberos-Datenbank unter Debian

Der Parameter `-s` muss auf jeden Fall hinter `create` geschrieben werden, sonst gibt es bei der Initialisierung eine Fehlermeldung.



Bei openSUSE liegt das Programm `kdb5_util`, genau wie alle weithin benötigten Kommandos, nicht im Pfad. Alle Programme, die Sie für die Verwaltung des KDCs benötigen, finden Sie im Verzeichnis `/usr/lib/mit/sbin/`. Am besten erweitern Sie Ihre Pfadeinstellungen um das Verzeichnis, wenn Sie nicht jedes Mal den gesamten Pfad eingeben wollen. Der Aufruf ist dann identisch mit dem auf einem Debian-System.

Wenn Sie CentOS einsetzen, dann befinden sich die Programme zwar im Pfad, aber der Aufruf ist anders. Auf einem CentOS-System initialisieren Sie die Datenbank mit dem Kommando `kdb5_util create -s`. Der Parameter `-s` muss nach `create` stehen, sonst erhalten Sie eine Fehlermeldung. Das Passwort wird verschlüsselt in der Datei `stash` abgelegt und befindet sich im jeweiligen Konfigurationsverzeichnis Ihrer verwendeten Distribution.

Erstellen der ACL-Datei

Damit die Zugriffe auf die Datenbank gesteuert werden können, müssen Sie vor dem ersten Test die Datei `kadm5.acl` noch anpassen.



Bei Debian und Ubuntu existiert die Datei nicht und muss von Ihnen erstellt werden. Über die ACLs wird der Zugriff für Benutzer und Dienste gesteuert. Achten Sie bei der Datei auf die unterschiedlichen Pfade der einzelnen Distributionen. Die Pfade für die Datei finden Sie in der Konfigurationsdatei `kdc.conf` der jeweiligen Distribution.

Unter CentOS und openSUSE sind die Dateien schon vorhanden. In der Datei auf einem openSUSE-System sind alle Zeilen auskommentiert, tragen Sie dort einfach Ihre Werte ein. Bei CentOS ist bereits eine aktive Zeile vorhanden, die aber in den meisten Fällen nutzlos ist oder im schlimmsten Fall eine Sicherheitslücke birgt. Entfernen Sie die Zeile oder kommentieren Sie sie zumindest aus.



Die Reihenfolge bei den ACLs beachten

Die ACLs werden der Reihe nach abgearbeitet. Sobald eine Regel auf eine Anfrage passt, wird die Abarbeitung der ACLs abgebrochen. Deshalb ist die Reihenfolge der ACLs besonders wichtig.

In Listing 14.8 sehen Sie eine einfache `kadm5.acl`-Datei:

```
*/admin@EXAMPLE.NET *
*@EXAMPLE.NET il
*/*@EXAMPLE.NET i
```

Listing 14.8 Inhalt der Datei »`kadm5.acl`«

Nach jeder Änderung an der Datei `kadm5.acl` starten Sie den KDC und den Admin-Server neu. Die einzelnen Einträge haben dabei folgende Bedeutungen:

- ▶ `*/admin@EXAMPLE.NET *`
Der Admin des KDC hat Vollzugriff auf die Datenbank.
- ▶ `*@EXAMPLE.NET il`
 - Mit diesem Recht ist es möglich, die Datenbank abzufragen (`i=info`).
 - Mit diesem Recht können die *Principals* ausgelesen werden (`l=list`). Jeder Benutzer kann immer das Passwort seines eigenen Principals ändern. Dazu bedarf es keiner zusätzlichen Rechte.

In Tabelle 14.2 sehen Sie eine kurze Übersicht über alle möglichen Rechte, die Sie über ACLs erteilen oder entziehen können. Ein kleiner Buchstabe erteilt die Rechte, ein großer Buchstabe entzieht die Rechte.

ACL-Flag	Rechte
a/A	Erlaubt oder verbietet das Hinzufügen eines Principals.
d/D	Erlaubt oder verbietet das Löschen eines Principals.
m/M	Erlaubt oder verbietet das Verändern eines Principals.
c/C	Erlaubt oder verbietet das Ändern von Passwörtern.
i/I	Erlaubt oder verbietet das Suchen in der Datenbank.
l/L	Erlaubt oder verbietet das Auflisten der Datenbank.

Tabelle 14.2 Liste der Kerberos-ACLs

Das Recht »c«

Gehen Sie sehr sparsam mit dem Recht `c` um. Ein Principal, der dieses Recht hat, kann alle Passwörter ändern.



14.4.1 Verwalten der Principals

Jetzt können Sie sich mit dem Admin-Server verbinden. Für die Verbindung zum Admin-Server gibt es zwei Programme: zum einen das Programm `kadmin.local` und zum anderen das Programm `kadmin`. Das Programm `kadmin.local` greift dabei direkt auf die Kerberos-Datenbank zu und benötigt selbst keine Authentifizierung.

Da im Moment noch gar kein Principal vorhanden ist, können Sie den ersten Kontakt zur Datenbank nur über das Kommando `kadmin.local` herstellen. Nachdem Sie alle benötigten Principals angelegt haben, verwenden Sie das Programm `kadmin`.



Nach dem Aufruf von `kadmin.local` befinden Sie sich in einem eigenen Prompt des Admin-Servers. Hier haben Sie jetzt die Möglichkeit, neue Principals anzulegen, bestehende zu ändern oder sich alle Principals auflisten zu lassen. In Listing 14.9 sehen Sie eine Liste aller bereits vorhandenen Principals:

```
root@kerberos01:/etc/krb5kdc# kadmin.local
Authentifizierung als Principal root/admin@EXAMPLE.NET mit Passwort
kadmin.local: listprincs
K/M@EXAMPLE.NET
kadmin/admin@EXAMPLE.NET
kadmin/changepw@EXAMPLE.NET
kadmin/kerberos01.example.net@EXAMPLE.NET
kiprop/kerberos01.example.net@EXAMPLE.NET
krbtgt/EXAMPLE.NET@EXAMPLE.NET
```

Listing 14.9 Liste der Standard-Principals



Ändern Sie keinen der Standard-Principals

Diese Principals, die Sie standardmäßig aufgelistet bekommen, werden zwingend so vom Kerberos-Server benötigt. Ändern Sie diese Principals nicht, da sonst der Kerberos-Server nicht mehr funktioniert.

Im nächsten Schritt werden jetzt die ersten Principals erzeugt. Achten Sie beim Erstellen der Principals auf Groß- und Kleinschreibung. Als Erstes legen Sie den `admin` für den Kerberos an. Anschließend können Sie weitere Principals anlegen. Hier sind sich alle Distributionen einig: Egal welche Distribution Sie nutzen, die Vorgehensweise ist immer identisch. In Listing 14.10 sehen Sie, wie die neuen Principals mit dem Kommando `addprinc` erzeugt werden:

```
kadmin.local: addprinc root/admin
Für root/admin@EXAMPLE.NET wurde keine Richtlinie angegeben, es wird die Vorgabe
»keine Richtlinie« verwandt.
Geben Sie das Passwort für Principal »root/admin@EXAMPLE.NET« ein.:
Geben Sie das Passwort für Principal »root/admin@EXAMPLE.NET« erneut ein.:
Principal "root/admin@EXAMPLE.NET" created.
```

```
kadmin.local: addprinc stefan
Für stefan@EXAMPLE.NET wurde keine Richtlinie angegeben, es wird die Vorgabe »keine
Richtlinie« verwandt.
Geben Sie das Passwort für Principal »stefan@EXAMPLE.NET« ein.:
Geben Sie das Passwort für Principal »stefan@EXAMPLE.NET« erneut ein.:
Principal "stefan@EXAMPLE.NET" created.
```

Listing 14.10 Erstellen der ersten Principals

Der erste Principal `root/admin` ist jetzt der Admin des Kerberos-Servers. Der zweite Principal `stefan` ist nur ein ganz normaler Benutzer. Wenn Sie im Anschluss noch einmal das Kommando `listprincs` verwenden, sehen Sie wie in Listing 14.11 Ihre neu angelegten Principals:

```
kadmin.local: listprincs
K/M@EXAMPLE.NET
kadmin/admin@EXAMPLE.NET
kadmin/changepw@EXAMPLE.NET
kadmin/kerberos01.example.net@EXAMPLE.NET
kiprop/kerberos01.example.net@EXAMPLE.NET
krbtgt/EXAMPLE.NET@EXAMPLE.NET
root/admin@EXAMPLE.NET
stefan@EXAMPLE.NET
```

Listing 14.11 Liste der neuen Principals

Im Gegensatz zu openSUSE und CentOS können Sie bei Debian und Ubuntu die *TAB-completion* in den Programmen `kadmin.local` und später auch in `kadmin` nutzen.

An dieser Stelle dürfen Sie das Programm `kadmin.local` noch nicht verlassen. Denn es reicht nicht aus, die Benutzer in der Datenbank anzulegen. Auch für den Client, von dem aus Sie auf den Kerberos-Server zugreifen wollen, benötigen Sie einen Principal, da sowohl eine Client- als auch eine Benutzerauthentifizierung bei der Anmeldung mit `kadmin` durchgeführt wird. Damit Sie sich auch vom Kerberos-Server selbst aus anmelden können, erstellen Sie noch, und zwar mit dem FQDN Ihres Servers, einen Principal in der Kerberos-Datenbank. Das machen Sie mit dem Kommando `addprinc -randkey host/kerberos01.example.net`. Dabei wird über den Parameter `-randkey` ein Schlüssel erzeugt, den der Client verwendet, um sich am Kerberos-Server zu authentifizieren. In Listing 14.12 sehen Sie den Vorgang:

```
kadmin.local: addprinc -randkey host/kerberos01.example.net
Für host/kerberos01.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwandt.
Principal "host/kerberos01.example.net@EXAMPLE.NET" created.
```

Listing 14.12 Erzeugen eines Host-Principals

LDAP nur auf Debian und Ubuntu

Da jetzt bereits bei CentOS und in Zukunft auch bei openSUSE der Dienst OpenLDAP nicht mehr bereitgestellt und unterstützt wird, werden wir die Einbindung in OpenLDAP nur noch am Beispiel von Debian/Ubuntu erklären.

Im nächsten Schritt sehen Sie, wie auch noch ein Principal für einen Service erstellt wird, der später auf einem Host verwendet werden soll. Als Beispiel verwenden wir hier den LD-



AP-Service. Im Verlauf dieses Kapitels werden wir Ihnen zeigen, wie Sie den Kerberos-Dienst in eine bestehende LDAP-Infrastruktur einbinden können. In einer Testumgebung können Sie später Kerberos und LDAP auf demselben System einrichten, aber in Produktivumgebungen sollten Sie die beiden Dienste aber auf getrennten Maschinen verwalten. Das erhöht die Sicherheit beider Dienste.

In Listing 14.13 sehen Sie, wie der Service erzeugt wird:

```
kadmin.local: addprinc -randkey ldap/ldap01.example.net
Für ldap/ldap01.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwendet.
Principal "ldap/ldap01.example.net@EXAMPLE.NET" created.
```

Listing 14.13 Erzeugung eines Hosts und eines Service

Der Parameter `-randkey` erstellt ein zufälliges Passwort für den Host und den Dienst. Da Hosts und Dienste kein Passwort eintippen können, sich aber auch authentifizieren müssen, wird dieses zufällige Passwort erzeugt und in der Datenbank abgelegt. Aus diesem zufälligen Passwort wird später eine *keytab*-Datei erzeugt, mit deren Hilfe sich der Host und der Dienst authentifizieren können. Ein `listprincs` zeigt Ihnen jetzt auch den neuen Host und den Service, die Sie erzeugt haben. Jetzt können Sie das Programm `kadmin.local` mit `exit` verlassen. Endlich ist es so weit: Sie können sich jetzt mit dem Principal `root/admin` am Kerberos authentifizieren und das Programm `kadmin` verwenden! In Listing 14.14 sehen Sie die Authentifizierung am Kerberos mit dem Programm `kinit`. Nach der Authentifizierung können Sie sich das vergebene Ticket mit dem Kommando `klist` auflisten lassen:

```
root@kerberos01:/etc/krb5kdc# kinit root/admin
Passwort für root/admin@EXAMPLE.NET:

root@kerberos01:/etc/krb5kdc# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: root/admin@EXAMPLE.NET

Valid starting      Expires            Service principal
08.12.2022 16:51:01  09.12.2022 02:51:01  krbtgt/EXAMPLE.NET@EXAMPLE.NET
                erneuern bis 09.12.2022 16:50:59

root@kerberos01:/etc/krb5kdc# kadmin
Authentifizierung als Principal root/admin@EXAMPLE.NET mit Passwort
Passwort für root/admin@EXAMPLE.NET:
```

Listing 14.14 Authentifizierung am Kerberos als »root«

Jeder Benutzer, der jetzt einen Principal in der Kerberos-Datenbank besitzt, kann sich mit dem Programm `kadmin` am Admin-Server anmelden. Aufgrund der vorher konfigurierten

ACLs hat er nur die Rechte, die Sie ihm eingeräumt haben. In Listing 14.15 sehen Sie, was passiert, wenn ein normaler Benutzer versucht, einen neuen Principal zu erstellen:

```
root@kerberos01:/etc/krb5kdc# kadmin -p stefan
Authentifizierung als Principal stefan mit Passwort
Passwort für stefan@EXAMPLE.NET:
kadmin: listprincs
K/M@EXAMPLE.NET
host/kerberos01.example.net@EXAMPLE.NET
kadmin/admin@EXAMPLE.NET
kadmin/changepw@EXAMPLE.NET
kadmin/kerberos01.example.net@EXAMPLE.NET
kiprop/kerberos01.example.net@EXAMPLE.NET
krbtgt/EXAMPLE.NET@EXAMPLE.NET
ldap/ldap01.example.net@EXAMPLE.NET
root/admin@EXAMPLE.NET
stefan@EXAMPLE.NET
kadmin: addprinc stka
Für stka@EXAMPLE.NET wurde keine Richtlinie angegeben, es wird die \
    Vorgabe »keine Richtlinie« verwendet.
Geben Sie das Passwort für Principal »stka@EXAMPLE.NET« ein.:
Geben Sie das Passwort für Principal »stka@EXAMPLE.NET« erneut ein.:
add_principal: Aktion erfordert »add«-Recht while creating "stka@EXAMPLE.NET".
```

```
kadmin: change_password -pw geheim stefan
Passwort von »stefan@EXAMPLE.NET« geändert
```

Listing 14.15 Zugriff auf »kadmin« durch einen normalen Benutzer

Sie sehen, dass der Benutzer zwar das Kommando zum Anlegen eines neuen Principals aufrufen kann, aber beim eigentlichen Anlegen des neuen Principals kommt es zu einer Fehlermeldung. Das Einzige, was der Benutzer hier machen kann, ist, sein Passwort zu ändern.

14.5 Kerberos und PAM

Nachdem der Kerberos-Server jetzt läuft, sollen sich die Benutzer über Kerberos an den entsprechenden Diensten authentifizieren können. Die meisten Linux-Dienste ermöglichen heute eine Authentifizierung über PAM, deshalb wird an dieser Stelle auch PAM für Kerberos konfiguriert werden. In diesem Abschnitt wird PAM nur für die Authentifizierung genutzt, aber nicht näher erläutert. Mehr zum Thema »PAM« finden Sie in Kapitel 5, »Berechtigungen«. Bei Debian und Ubuntu benötigen Sie keine Änderung mehr an den Dateien in */etc/pam.d*. Das PAM-Modul wird automatisch in alle *common*-*-Dateien eingetragen. Wenn Sie beim Installieren des Betriebssystems einen lokalen Benutzer angelegt

haben und Sie nicht wollen, dass der Benutzer sich über Kerberos authentifizieren muss, dann passen Sie die *common*-*-Dateien so an, dass erst ab der UID 1001 Kerberos genutzt wird. Dadurch kann sich der Benutzer auch dann noch anmelden, wenn der Kerberos-Dienst einmal nicht erreichbar ist.

14.5.1 Konfiguration der PAM-Dateien auf einem openSUSE-System



Achten Sie beim Ändern der PAM-Einstellungen immer darauf, dass Sie noch mindestens eine Konsole geöffnet haben, an der der Benutzer *root* angemeldet ist, denn Änderungen an den PAM-Dateien sind sofort nach dem Speichern wirksam. Sollten Sie einen Fehler in der Datei haben, kann es Ihnen passieren, dass Sie sich nicht mehr anmelden können.

Bei openSUSE fügen Sie das entsprechende PAM-Modul selbst in die Konfiguration ein. Das geschieht mit dem Kommando `pam-config`. Das Kommando `pam-config` verändert im Verzeichnis `/etc/pam.d` nur die *common*-*-*pc*-Dateien. Auf diese zeigt dann der jeweilige Link der *common*-*-Dateien. In Listing 14.16 sehen Sie, wie die entsprechenden Einträge zu den PAM-Dateien hinzugefügt werden:

```
kerberos:/etc/pam.d # pam-config --add --krb5 --krb5-minimum_uid=1001
```

Listing 14.16 Konfiguration von PAM mittels »pam-config«



Lokalen Benutzern weiterhin die Anmeldung erlauben

Wenn Sie den Parameter `--krb5-minimum_uid=1001` vergessen, kann sich kein lokaler Benutzer mehr ohne Kerberos-Principal anmelden. Das betrifft auch den Benutzer *root*.

Wie Sie sehen, haben wir nicht die UID=1000 als erste UID gewählt, sondern die UID=1001. So kann sich der bei der Installation angelegte Benutzer auch noch am System anmelden.



Sollten Sie bei der Anmeldung auf Probleme stoßen, können Sie in der Datei *common-auth* hinter den Modulen `pam_unix2` und `pam_krb5` die Option `debug` setzen. Dann können Sie in der Datei `/var/log/messages` sehr leicht einen Fehler in der PAM-Konfiguration aufspüren.

Leider müssen wir uns an dieser Stelle von CentOS verabschieden, denn mit CentOS 8, also auch mit der von uns verwendeten CentOS-Stream-Version, ist das Paket *pam_krb5* als *deprecated* gekennzeichnet und wird nicht mehr bereitgestellt.

14.5.2 Testen der Anmeldung

Wenn Sie jetzt für einen vorhandenen Benutzer auf Ihrem System einen Principal in der Kerberos-Datenbank erzeugt haben, erhält der Benutzer bei der Anmeldung gleich ein TGT vom Kerberos-Server. Legen Sie sich einen Benutzer auf dem System an, aber vergeben Sie

kein Passwort an den Benutzer. Erzeugen Sie anschließend einen Kerberos-Principal für den neuen Benutzer, und melden Sie sich dann via *ssh* oder lokal am System mit dem neuen Benutzer und dem Passwort des Principals an. In Listing 14.17 sehen Sie eine Anmeldung eines bestehenden Benutzers über *ssh*. Direkt nach der Anmeldung wird mit *klist* das TGT abgefragt:

```
ssh stefan@192.168.56.55
stefan@192.168.56.55's password:
Linux kerberos01

stefan@kerberos01:~$ klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_1112_bfVgfv
Standard-Principal: stefan@EXAMPLE.NET

Valid starting      Expires            Service principal
08.12.2022 17:30:49 09.12.2022 03:30:49 krbtgt/EXAMPLE.NET@EXAMPLE.NET
                erneuern bis 09.12.2022 17:30:49
```

Listing 14.17 Anmeldung mit Kerberos

14.6 Neue Benutzer mit Kerberos-Principal anlegen

Wenn Sie jetzt einen neuen Benutzer in Ihrem System anlegen, geschieht das in zwei Schritten. Im ersten Schritt erzeugen Sie mit dem Programm *kadmin* einen Principal für den Benutzer. Im zweiten Schritt legen Sie dann wie gewohnt den Benutzer mit *useradd* an. Die Vergabe eines Passworts mit *passwd* ist nicht mehr notwendig. Für die Anmeldung am System wird das Kerberos-Passwort verwendet. In Listing 14.18 sehen Sie alle Schritte für das Anlegen eines neuen Benutzers:



```
root@kerberos01:~# kadmin
Authentifizierung als Principal root/admin@EXAMPLE.NET mit Passwort
Passwort für root/admin@EXAMPLE.NET:
kadmin: addprinc ktom
Für ktom@EXAMPLE.NET wurde keine Richtlinie angegeben, es wird die \
    Vorgabe »keine Richtlinie« verwandt.
Geben Sie das Passwort für Principal »ktom@EXAMPLE.NET« ein.:
Geben Sie das Passwort für Principal »ktom@EXAMPLE.NET« erneut ein.:
Principal "ktom@EXAMPLE.NET" created.
kadmin: quit

root@kerberos01:~# useradd -m -s /bin/bash ktom
```

Listing 14.18 Anlegen eines neuen Benutzers

Der Benutzer `ktom`, der gerade angelegt wurde, hat in der Datei `/etc/shadow` den in Listing 14.19 aufgeführten Eintrag:

```
ktom:!:19334:0:99999:7:::
```

Listing 14.19 Eintrag eines neuen Benutzers in der Datei `»/etc/shadow«`

Sie sehen: Hier wird kein Passwort mehr verwaltet. Eine Änderung des Passworts für den Benutzer `ktom` finden Sie in Listing 14.20. Es wird jetzt nur noch das Kerberos-Passwort geändert. Der Eintrag in `/etc/shadow` ändert sich nicht.

```
root@kerberos01:~# passwd ktom
Current Kerberos password:
Enter new Kerberos password:
Retype new Kerberos password:
passwd: Passwort erfolgreich geändert
```

Listing 14.20 Änderung des Passworts für einen Benutzer

14.7 Hosts und Dienste

Im letzten Abschnitt haben wir uns um die Benutzer gekümmert, die sich über Kerberos anmelden sollen. Jetzt benötigen noch die Hosts und die Services die Möglichkeit, sich über Kerberos zu authentifizieren. Das ist notwendig, da Kerberos, zusätzlich zur Benutzer-authentifizierung, noch eine Host- und Service-Authentifizierung verwendet.

Ein Host oder ein Dienst ist ja nicht in der Lage, sein Passwort einzugeben, also muss das Passwort für den Host oder den Service anders abgeglichen werden. Dieser Vorgang läuft über die *keytab*-Dateien. Die Standard-*keytab*-Datei für das System ist die Datei `/etc/krb5.keytab`. In dieser Datei können Sie auch alle Schlüssel für alle Dienste verwalten, die auf diesem Host laufen. Besser ist es aber, wenn Sie für jeden Dienst eine eigene *keytab*-Datei verwalten. Da die *keytab*-Datei immer auf dem System liegen muss, das auch den Dienst bereitstellt, ist es aus Sicherheitsgründen besser, die Dateien separat zu verwalten und auf dem System nur mit den benötigten Rechten auszustatten. Dann ist es auch einfacher, den Schlüssel zu entfernen, wenn Sie einmal einen Dienst von einem Host dauerhaft entfernen wollen. Beim Erstellen der Principals in Abschnitt 14.4.1, »Verwalten der Principals«, haben Sie ja schon einen Principal für einen Host und einen Service erstellt. Für diese beiden Principals wird jetzt eine eigene *keytab*-Datei erzeugt. Dazu verwenden Sie wieder das Programm `kadmin`. Nachdem Sie sich an `kadmin` angemeldet haben, können Sie den neuen Eintrag in der *keytab*-Datei so erstellen, wie in Listing 14.21 gezeigt:

```
kadmin: addprinc -randkey host/ldap01.example.net
Für host/ldap01.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwandt.
Principal "host/ldap01.example.net@EXAMPLE.NET" created.
```



```
admin: ktadd -k /etc/krb5.keytab host/kerberos01.example.net
Der Eintrag für Principal host/kerberos01.example.net mit KVNO 2 \
  und Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
  Schlüsseltabelle WRFIL:etc/krb5.keytab hinzugefügt.
Der Eintrag für Principal host/kerberos01.example.net mit KVNO 2 \
  und Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
  Schlüsseltabelle WRFIL:etc/krb5.keytab hinzugefügt.
```

```
kadmin: ktadd -k /root/ldap01-krb5.keytab host/ldap01.example.net
Der Eintrag für Principal host/ldap01.example.net mit KVNO 2 und \
  Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
  Schlüsseltabelle WRFIL:/root/ldap01-krb5.keytab hinzugefügt.
Der Eintrag für Principal host/ldap01.example.net mit KVNO 2 und \
  Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
  Schlüsseltabelle WRFIL:/root/ldap01-krb5.keytab hinzugefügt.
```

```
kadmin: ktadd -k /root/ldap01-ldap.keytab ldap/ldap01.example.net
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 und \
  Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
  Schlüsseltabelle WRFIL:/root/ldap01-ldap.keytab hinzugefügt.
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 und \
  Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
  Schlüsseltabelle WRFIL:/root/ldap01-ldap.keytab hinzugefügt.
```

Listing 14.21 Eintrag in die »keytab«-Datei

Auf diese Art und Weise können Sie jetzt alle benötigten Dienste und Server in einer eigenen *keytab*-Datei speichern. In den Beispielen haben wir erst einen neuen Hostprincipal für den LDAP-Server erzeugt und dann die *keytab*-Dateien für den ersten Kerberos-Server, den ersten LDAP-Server und den LDAP-Dienst auf dem ersten LDAP-Server. Die beiden *keytab*-Dateien für den LDAP-Server werden später auf den entsprechenden LDAP-Server kopiert.

Achten Sie auf die »KVNO«

Bei der *Key Version Number* (KVNO) handelt es sich um die aktuelle Seriennummer eines Eintrags in einer *keytab*-Datei. Bei der Authentifizierung eines Dienstes oder eines Hosts wird die KVNO immer mit der Version in der Datenbank verglichen. Nur wenn die Versionsnummer identisch ist, ist eine Authentifizierung möglich. Wenn eine Authentifizierung fehlschlägt, prüfen Sie immer auch die KVNO.

Die Kommandos zum Erstellen der *keytab*-Dateien sind bei allen Distributionen identisch, nur die Art und Anzahl der erstellten Schlüssel in der Datei können variieren. Um die Version abzugleichen, benötigen Sie die Information zur KVNO aus der Datenbank und die KVNO



aus der *keytab*-Datei des Hosts oder des Service. Zur Ermittlung der KVNO in der Datenbank verwenden Sie das Kommando `kvno` so wie in Listing 14.22:

```
root@kerberos01:~# kvno host/kerberos01.example.net
host/kerberos01.example.net@EXAMPLE.NET: KVNO = 2
```

```
root@kerberos01:~# kvno host/ldap01.example.net
host/ldap01.example.net@EXAMPLE.NET: KVNO = 2
```

```
root@kerberos01:~# kvno ldap/ldap01.example.net
ldap/ldap01.example.net@EXAMPLE.NET: KVNO = 2
```

Listing 14.22 Ermittlung der KVNO in der Datenbank

Sie sehen hier, dass Sie die KVNO für alle Hosts, Services und alle Benutzer einzeln abrufen können. Jetzt benötigen Sie noch die Information der entsprechenden *keytab*-Dateien.

Da Sie für Benutzer in den seltensten Fällen eine *keytab*-Datei erzeugen (ein Benutzer sollte sein Passwort immer eintippen), sehen Sie in Listing 14.23 nur die Auswertung der Standard-*keytab*-Datei und der Service-*keytab*-Datei:

```
root@kerberos01:~# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
-----
  2 host/kerberos01.example.net@EXAMPLE.NET
  2 host/kerberos01.example.net@EXAMPLE.NET
root@kerberos01:~# klist -k /root/ldap01-krb5.keytab
Keytab name: FILE:/root/ldap01-krb5.keytab
KVNO Principal
-----
  2 host/ldap01.example.net@EXAMPLE.NET
  2 host/ldap01.example.net@EXAMPLE.NET
root@kerberos01:~# klist -k /root/ldap01-ldap.keytab
Keytab name: FILE:/root/ldap01-ldap.keytab
KVNO Principal
-----
  2 ldap/ldap01.example.net@EXAMPLE.NET
  2 ldap/ldap01.example.net@EXAMPLE.NET
```

Listing 14.23 Ermitteln der KVNO aus den »keytab«-Dateien

Ein anderer Weg, wie Sie sich den Inhalt der *keytab*-Datei auflisten lassen können, ist die Verwendung des Programms `ktutil`. Nach dem Aufruf des Programms laden Sie als Erstes die *keytab*-Datei, die Sie sich ansehen wollen. Verwenden Sie dazu das Kommando `rkt /etc/krb5.keytab`. In Listing 14.24 sehen Sie den Inhalt der *keytab*-Datei:

```

root@kerberos01:~# ktutil
ktutil: rkt /etc/krb5.keytab
ktutil: l
slot KVNO Principal
-----
 1  2  host/kerberos01.example.net@EXAMPLE.NET
 2  2  host/kerberos01.example.net@EXAMPLE.NET

```

Listing 14.24 Inhalt der »keytab«-Datei

Jetzt können Sie sich auch mithilfe der *keytab*-Datei authentifizieren, wenn Sie das Programm *kadmin* starten. In Listing 14.25 sehen Sie ein Beispiel dafür:

```

root@k1-stat:~# kadmin -k
Authentifizierung als Principal host/kerberos01.example.net@EXAMPLE.NET \
  mit Standardschlüsseltabelle
kadmin: addprinc ptau
WARNUNG: Für ptau@EXAMPLE.NET wurde keine Richtlinie angegeben, \
  es wird die Vorgabe »keine Richtlinie« verwandt.
Geben Sie das Passwort für Principal »ptau@EXAMPLE.NET« ein.:
Geben Sie das Passwort für Principal »ptau@EXAMPLE.NET« erneut ein.:
add_principal: Aktion erfordert »add«-Recht while creating "ptau@EXAMPLE.NET".

```

Listing 14.25 Authentifizierung mit der »keytab«-Datei

Wie Sie in dem Beispiel sehen, können Sie so aber keine neuen Principals erzeugen, da der Host keine Rechte dazu besitzt. Wollen Sie über diese Art der Authentifizierung auch neue Principals erzeugen, passen Sie die ACLs entsprechend an. Denken Sie jedoch dabei daran, dass dann jeder Benutzer, der auf diesem System angemeldet ist, auch Principals verwalten kann. Besser ist es daher, sich immer über *kadmin* mit einem Passwort zu authentifizieren.

14.7.1 Einträge entfernen

Irgendwann wollen Sie mit Sicherheit auch einmal einen Benutzer, einen Host oder einen Service aus der Kerberos-Datenbank entfernen. Um nun einen Benutzer aus dem System komplett zu entfernen, gehen Sie so vor wie in Listing 14.26:

```

root@k1-stat:~# kadminkadmin: delprinc ktom
Sind Sie sicher, dass Sie den Principal »ktom@EXAMPLE.NET« \
  löschen möchten? (yes/no): yes
Principal »ktom@EXAMPLE.NET« gelöscht
Stellen Sie sicher, dass Sie diesen Principal aus allen \
  ACLs entfernt haben, bevor Sie ihn erneut benutzen.

```

Listing 14.26 Löschen eines Benutzers

Jetzt können Sie den Benutzer mit `userdel` aus dem System entfernen. Wenn Sie einen Host oder einen Service entfernen möchten, entfernen Sie auf jeden Fall neben dem Principal auch immer noch den Eintrag aus der `keytab`-Datei. Wenn Sie für alle Services eine eigene `keytab`-Datei erzeugt haben, löschen Sie einfach die entsprechende Datei. Haben Sie alle Schlüssel in derselben Datei eingetragen, sehen Sie in Listing 14.27, wie Sie einen einzelnen Service entfernen können. Genauso gehen Sie auch für einen Host vor.

```
root@kerberos01:~# kadmin
kadmin: ktrem -k /root/ldap.keytab ldap/ldap01.example.net
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 \
wurde aus der Schlüsseltabelle WRFILE:/etc/ldap.keytab entfernt.
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 \
wurde aus der Schlüsseltabelle WRFILE:/etc/ldap.keytab entfernt.

kadmin: delprinc ldap/ldap01.example.net
Sind Sie sicher, dass Sie den Principal »ldap/ldap01.example.net@EXAMPLE.NET« \
löschen möchten? (yes/no): y
Principal »ldap/ldap01.example.net@EXAMPLE.NET« nicht gelöscht

kadmin: delprinc ldap/ldap01.example.net
Sind Sie sicher, dass Sie den Principal »ldap/ldap01.example.net@EXAMPLE.NET« \
löschen möchten? (yes/no): yes
Principal »ldap/ldap01.example.net@EXAMPLE.NET« gelöscht
Stellen Sie sicher, dass Sie diesen Principal aus allen ACLs entfernt haben, \
bevor Sie ihn erneut benutzen.
```

Listing 14.27 Einen Service entfernen



Sie sehen in diesem Listing, dass der erste Versuch, den Principal aus der Datenbank zu löschen, fehlgeschlagen ist, da bei der Frage, ob der Principal gelöscht werden soll, nur ein `y` eingegeben wurde. Erst beim zweiten Versuch mit der Eingabe `yes` wurde der Principal gelöscht. Jetzt sind Sie in der Lage, die Kerberos-Datenbank zu verwalten. In den einzelnen Kapiteln zu den verschiedenen Diensten beschreiben wir, wie Sie den jeweiligen Dienst zusammen mit Kerberos nutzen.


14.8 Konfiguration des Kerberos-Clients

Nachdem der Kerberos-Server läuft, soll in diesem Abschnitt nun der Kerberos-Client konfiguriert werden. Jeder Host in Ihrem Netzwerk, der eine Authentifizierung über Kerberos durchführen soll, benötigt eine entsprechende Konfiguration.



Denken Sie daran, dass die Authentifizierung über Kerberos sehr zeitkritisch ist. Aus diesem Grund ist es sehr sinnvoll, wenn in Ihrem Netzwerk ein Zeitserver läuft.

Ein weiterer Punkt ist die Namensauflösung. Wenn Sie noch keinen Nameserver für Ihr Netzwerk laufen haben, ist jetzt der richtige Zeitpunkt, einen DNS-Server einzurichten, der sowohl den Kerberos-Server als auch alle Kerberos-Clients ordnungsgemäß auflösen kann, und zwar sowohl vorwärts als auch rückwärts. Im ersten Schritt installieren Sie die Pakete für die Verwaltung des Clients auf den Hosts. Bei Debian und Ubuntu sind das die Pakete `krb5-config` und `krb5-user`. Für openSUSE installieren Sie das Paket `krb5-client` über den YaST. Während der Installation der Pakete werden Sie nach dem Admin-Server und dem Kerberos-Server gefragt. Geben Sie dort den FQDN Ihres Kerberos-Servers an. Die Eingaben werden direkt in die Datei `/etc/krb5.conf` übernommen. Da wir aber im letzten Abschnitt den Kerberos-Server bereits installiert haben, der ja auch schon die Clientkonfiguration in der Datei `/etc/krb5.conf` erhalten hat, können Sie an dieser Stelle die Datei einfach vom Server auf den Client kopieren. Die Datei `/etc/krb5.conf` kann auf allen Clients identisch sein.

Denken Sie daran, für jeden Client einen Principal in Ihrer Kerberos-Datenbank zu erzeugen, denn ohne einen Eintrag in der Datenbank kann der Client keine Authentifizierung eines Benutzers durchführen. 


Damit ist die Konfiguration des Clients abgeschlossen. Jetzt können Sie die Konfiguration des Clients testen, indem Sie mit `kinit` ein TGT für einen Benutzer holen. In Listing 14.28 sehen Sie, wie das TGT vom Kerberos-Server für einen lokalen Benutzer auf dem Client geholt wird:

```
root@client-01:~# kinit stefan
Passwort für stefan@EXAMPLE.NET:
```

```
root@client-01:~# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: stefan@EXAMPLE.NET
```

```
Valid starting      Expires            Service principal
08.12.2022 19:11:23 09.12.2022 05:11:23 krbtgt/EXAMPLE.NET@EXAMPLE.NET
    erneuern bis 09.12.2022 19:11:21
```

Listing 14.28 Kerberos-Authentifizierung am Client

Zu diesem Zeitpunkt gibt es keine zentrale Benutzerverwaltung. Legen Sie den Benutzer also lokal an, um sich ein TGT vom Kerberos-Server holen zu können. In Kapitel 17, »LDAP«, wird beschrieben, wie Sie eine zentrale Benutzerverwaltung mit LDAP realisieren können. In Abschnitt 14.11 »Kerberos in LDAP einbinden« werden wir Ihnen zeigen, wie Sie Kerberos mit LDAP verbinden können. 

14.8.1 PAM und Kerberos auf dem Client

Damit sich die Benutzer nun auch direkt über Kerberos authentifizieren können, benötigt der Client noch eine PAM-Konfiguration. Dazu gehen Sie genauso vor wie zuvor schon auf

dem Kerberos-Server. Installieren Sie als Erstes die entsprechenden PAM-Pakete für Ihre Distribution auf allen Clients. Erstellen Sie anschließend ein lokales Benutzerkonto auf dem Client, für das es auf dem Kerberos-Server einen Principal gibt. Achten Sie darauf, dass der Benutzer dieselbe UID hat wie der Benutzer auf dem Server, da es sonst zu Problemen kommt, wenn Sie zusätzliche Dienste, wie zum Beispiel NFS, konfigurieren wollen. Ein Passwort müssen Sie für den neuen Benutzer nicht vergeben, denn die Authentifizierung findet ja jetzt über Kerberos statt. Alle Tickets, die ein Benutzer besitzt, kann er mit dem Kommando `kdestroy` selbst entfernen.



Achtung bei openSUSE-Clients

Beachten Sie bitte in jedem Fall bei allen openSUSE-Clients die Problematik mit der PAM-Konfiguration aus Abschnitt 14.5.1, »Konfiguration der PAM-Dateien auf einem openSUSE-System«.

14.9 Replikation des Kerberos-Servers

Der Kerberos-Server ist jetzt die zentrale Authentifizierungsquelle in Ihrem Netzwerk. Wenn Sie nur einen Kerberos-Server haben, können Ihre Anwender die Dienste nicht mehr nutzen, wenn der Kerberos-Server einmal ausfallen sollte. Aus diesem Grund ist es sinnvoll, dass Sie immer mindestens zwei Kerberos-Server in Ihrem Netzwerk haben. Es gibt verschiedene Wege, um die Replikation zwischen dem KDC-Master und den KDC-Slaves durchzuführen. Der wohl beste Weg ist die Verwendung des Programms `kprop` zur Replikation der Datenbank von einem Server auf einen anderen. Dieser Vorgang wird *Propagation* genannt. Genau diese Art der Replikation wird an dieser Stelle beschrieben.

Die elegantere, aber etwas aufwendigere Methode über eine zentrale Principal-Datenbank im LDAP werden wir Ihnen in Abschnitt 14.11 »Kerberos in LDAP einbinden« vorstellen.

14.9.1 Bekanntmachung aller KDCs im Netz

In diesem Abschnitt geht es darum, den schon laufenden KDC auf einen zweiten Server zu replizieren, der als KDC-Slave läuft. Für die Replikation müssen alle KDC-Slaves wissen, wo ihr KDC-Master ist. Dafür gibt es zwei Möglichkeiten. Die erste ist die einfachere, die immer dann ausreicht, wenn nur ein zusätzlicher KDC-Slave eingerichtet werden soll und auch in Zukunft nicht unbedingt weitere KDC-Slaves benötigt werden. Für diese Art der Replikation werden einfach alle KDCs über die Datei `krb5.conf` bekannt gemacht.

Die zweite Möglichkeit ist die Bekanntmachung der KDCs über *SRV-Einträge* im DNS-Server. Das ist immer dann die richtige Wahl, wenn Sie mehrere KDC-Slaves einrichten wollen. Dabei benötigen Sie natürlich Zugriff auf die Zonendateien Ihres DNS-Servers. Beide Varianten sollen hier vorgestellt werden.

Bekanntmachung aller KDCs über die Datei »krb5.conf«

Um alle Server bekannt zu machen, passen Sie lediglich den Bereich [realms] so wie in Listing 14.29 an:

```
[realms]
    EXAMPLE.NET = {
        kdc = kerberos01.example.net
        kdc = kerberos02.example.net
        admin_server = kerberos01.example.net
    }
```

Listing 14.29 Anpassung der Datei »krb5.conf«

Der Rechner mit dem FQDN `kerberos02.example.net` wird als KDC-Slave eingerichtet. Wie Sie sehen, gibt es in der Konfiguration nur einen `admin_server`. Das ist auch richtig so, denn alle Änderungen werden immer am KDC-Master durchgeführt. Die KDC-Slaves dienen nur zur Authentifizierung, sind also *Read-only*-KDCs. Die geänderte *krb5.conf* legen Sie jetzt auf allen Clients und auf allen Ihren KDC-Slaves an. Damit ist die einfache Art der Bekanntgabe aller KDCs auch schon abgeschlossen. Wenn Sie diese Art der Bekanntmachung einsetzen, dann denken Sie daran, dass bei jeder Änderung der KDCs auch immer die *krb5.conf* auf allen Kerberos-Clients eine Anpassung benötigt.

14

Bekanntmachung aller KDCs über SRV-Einträge im DNS

Etwas aufwendiger, aber dafür nur einmalig durchzuführen ist die Möglichkeit, die KDCs in Ihrem Netzwerk über SRV-Einträge im DNS bekannt zu machen. Sollten Sie dann später weitere KDCs in Ihrem Netz benötigen, reicht es, diese einfach auf Ihrem DNS-Server in der entsprechenden Zone einzutragen. In der Forward-Zonendatei Ihrer DNS-Zone tragen Sie zusätzlich die in Listing 14.30 aufgeführten Einträge ein:

```
; srv- und txt-Einträge für die Kerberos-Server
; REALM-Registrierung
_kerberos      IN      TXT      EXAMPLE.NET

; Port 88 für den KDC, UDP und TCP für Kerberos5
_kerberos._udp IN SRV 01 00 88 kerberos01.example.net.
_kerberos._tcp IN SRV 01 00 88 kerberos01.example.net.

; Der KDC-Master
_kerberos-master._udp IN SRV 01 00 88 kerberos01.example.net.
_kerberos-master._tcp IN SRV 01 00 88 kerberos01.example.net.

; Port 749, für den kadmin aber nur TCP
_kerberos-adm._tcp IN SRV 01 00 749 kerberos01.example.net.
```

```
; Port 464 für die Verwendung von kpasswd, nur UDP
_kpasswd._udp IN SRV 01 00 464 kerberos01.example.net.
```

```
; Port 88, für alle KDC-Slaves
_kerberos._udp IN SRV 01 00 88 kerberos02.example.net.
_kerberos._tcp IN SRV 01 00 88 kerberos02.example.net.
```

Listing 14.30 SRV- und TXT-Einträge im DNS

Die Einträge haben folgende Bedeutungen:

- ▶ `_kerberos IN TXT EXAMPLE.NET`
Durch diesen TXT-Record wird der Kerberos-Realm der DNS-Zone zugeordnet.
- ▶ `_kerberos._udp IN SRV 01 00 88 kerberos01.example.net.`
Der TCP- und UDP-Port 88 des KDC-Master wird für alle Clients bekannt gemacht.
- ▶ `_kerberos-master._udp IN SRV 01 00 88 kerberos01.example.net.`
Dieser Eintrag zeigt immer auf den KDC, der für die Verwaltung der Passwörter verantwortlich ist. Wenn ein Benutzer bei der Anmeldung anscheinend ein falsches Passwort angibt, wird der Anmeldeprozess an diesen KDC weitergeleitet, und dort wird die Anmeldung erneut geprüft. Erst wenn auch bei diesem KDC die Anmeldung fehlschlägt, erhält der Benutzer eine Fehlermeldung. Sollte also ein Benutzer sein Passwort ändern, sich sofort wieder anmelden und sollte dann die Anmeldung an einem KDC geprüft werden, der noch nicht synchronisiert ist, so wird die Anmeldung trotzdem mit dem neuen Passwort durchgeführt.
- ▶ `_kerberos-adm._tcp IN SRV 01 00 749 kerberos01.example.net.`
Hier wird der TCP-Port 749 bekannt gegeben, über den der Admin-Server für den Kerberos erreichbar ist. Dieser Eintrag darf nur auf den KDC-Master zeigen, da hier alle Einträge verwaltet werden.
- ▶ `_kpasswd._udp IN SRV 01 00 464 kerberos01.example.net.`
Der UDP-Port 464 wird immer dann benötigt, wenn ein Benutzer sein Passwort ändert. Der Eintrag darf auch nur auf den KDC-Master zeigen.
- ▶ `_kerberos._udp IN SRV 01 00 88 kerberos02.example.net.`
Einträge der KDC-Slaves. Da die Slaves nur für die Authentifizierung verantwortlich sind, muss hier nur der TCP- und UDP-Port 88 bekannt gemacht werden.

Nachdem Sie alle Einträge in der Zonendatei erstellt und den DNS-Server neu gestartet haben, können Sie die SRV-Einträge jetzt so wie in Listing 14.31 prüfen:

```
root@kerberos01:~# host -t srv _kerberos-master._tcp.example.net
_kerberos-master._tcp.example.net has SRV record 1 0 88 kerberos01.example.net.
```

```
root@kerberos01:~# host -t srv _kerberos-adm._tcp.example.net
_kerberos-adm._tcp.example.net has SRV record 1 0 749 kerberos01.example.net.
```



```

root@kerberos01:~# host -t srv _kpasswd._udp.example.net
_kpasswd._udp.example.net has SRV record 1 0 464 kerberos01.example.net.

root@kerberos01:~# host -t srv _kerberos._tcp.example.net
_kerberos._tcp.example.net has SRV record 1 0 88 kerberos01.example.net.
_kerberos._tcp.example.net has SRV record 1 0 88 kerberos02.example.net.

```

Listing 14.31 Testen der SRV-Einträge

Wenn alle SRV-Einträge aufgelöst werden, können Sie jetzt die Datei *krb5.conf* so anpassen, wie es in Listing 14.32 dargestellt ist:

```

[libdefaults]
    default_realm = EXAMPLE.NET

[realms]
    EXAMPLE.NET = {
        admin_server = kerberos01.example.net
    }

[domain_realm]
    .example.net = EXAMPLE.NET

[logging]
    kdc = FILE:/var/log/krb5/krb5kdc.log
    admin_server = FILE:/var/log/krb5/kadmind.log
    default = SYSLOG:NOTICE:DAEMON

```

Listing 14.32 Angepasste »krb5.conf«

Wie Sie sehen, sind in der Datei keine Einträge mehr für KDCs vorhanden. Lediglich der Eintrag *admin_server* bleibt erhalten. Da über den SRV-Eintrag nicht alle Funktionen des Admin-Servers abgedeckt werden können, muss der Eintrag auch bestehen bleiben. Jetzt können Sie beliebig viele KDC-Slaves einrichten und brauchen nur noch den DNS-Server anzupassen und nicht mehr alle *krb5.conf*-Dateien. Kopieren Sie einfach diese Datei auf alle Clients und Server, die sich über Kerberos authentifizieren sollen.

Leider unterstützt der MIT-Kerberos den SRV-Record für den Kerberos-Admin-Server nicht. Aus diesem Grund muss der Admin-Server auch weiterhin in der Datei */etc/krb5.conf* eingetragen sein.

**14.9.2 Konfiguration des KDC-Masters**

Als Erstes konfigurieren Sie den KDC-Master, denn nur dann kann der Slave die Daten vom Master replizieren. Dazu erstellen Sie für jeden KDC-Slave einen Principal. Im selben Schritt

erstellen Sie die *krb5.keytab*-Dateien für den KDC-Master (sofern Sie die Datei noch nicht erstellt haben) und die KDC-Slaves. Das Anlegen des Principals führen Sie mit dem Programm *kadmin* wie in Listing 14.33 aus:

```
root@k1-stat:~# kadmin
Authentifizierung als Principal root/admin@EXAMPLE.NET mit Passwort
Passwort für root/admin@EXAMPLE.NET:

kadmin: addprinc -randkey host/kerberos02.example.net
WARNUNG: Für host/kerberos02.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwendet.
Principal "host/kerberos02.example.net@EXAMPLE.NET" created.
```

Listing 14.33 Erzeugen des Principals für den KDC-Slave

Im nächsten Schritt erzeugen Sie die entsprechende *krb5.keytab*-Datei für die Slave-KDCs. Achten Sie darauf, dass Sie hier auf jeden Fall einen neuen Dateinamen angeben, denn sonst wird der Schlüssel in die *keytab*-Datei des Masters geschrieben. Diese Datei kopieren Sie dann auf den Slave-KDC in das Verzeichnis */etc*. Die Datei wird immer auf dem KDC-Master erzeugt und dann auf die KDC-Slaves kopiert. Die Datei erzeugen Sie mit dem Befehl *ktadd* innerhalb des Programms *kadmin*. Wenn Sie bei dem Befehl keine Datei mit der Option *-k* angeben, werden die Schlüssel immer in die Datei */etc/krb5.keytab* geschrieben. Das wäre auch möglich, dann hätten Sie aber immer alle Schlüssel für alle KDCs auf allen KDCs. Besser ist es, wenn jeder KDC lediglich seinen eigenen Schlüssel besitzt. In Listing 14.34 sehen Sie, wie die Datei mit dem KDC-Master-Key und dem KDC-Slave-Key erzeugt wird:

```
kadmin: listprincs
K/M@EXAMPLE.NET
host/kerberos01.example.net@EXAMPLE.NET
host/kerberos02.example.net@EXAMPLE.NET
kadmin/admin@EXAMPLE.NET
kadmin/changepw@EXAMPLE.NET
kadmin/kerberos01.example.net@EXAMPLE.NET
kiprop/kerberos01.example.net@EXAMPLE.NET
krbtgt/EXAMPLE.NET@EXAMPLE.NET
ldap/ldap01.example.net@EXAMPLE.NET
root/admin@EXAMPLE.NET
stefan@EXAMPLE.NET

kadmin: ktadd -k /root/kerberos02.keytab host/kerberos02.example.net
Der Eintrag für Principal host/k2-stat.example.net mit KVNO 2 \
    und Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
    Schlüsseltabelle WRFILE:/root/kerberos02.keytab hinzugefügt.
Der Eintrag für Principal host/kerberos02.example.net mit KVNO 2 \
```

und Verschlüsselungstyp `aes128-cts-hmac-sha256-128` wurde der \ Schlüsseltabelle `WRFILE:/root/kerberos02.keytab` hinzugefügt.

Listing 14.34 Erzeugung der »keytab«-Dateien

Bevor die Datei für den ersten Slave auf den zweiten Kerberos-Server kopiert wird, lassen Sie sich den Inhalt der Datei noch einmal auflisten. Denken Sie daran: Es handelt sich dabei um eine Datei mit den verschlüsselten Passwörtern. Sie können diese Datei daher nicht mit einem Editor öffnen und sollten das auch nicht versuchen. Listing 14.35 zeigt Ihnen, wie Sie sich den Inhalt der Datei mit dem Kommando `ktutil` anzeigen lassen können:

```
root@kerberos01:~# ktutil
ktutil: rkt /etc/krb5.keytab
ktutil: l
slot KVNO Principal
-----
 1  2  host/kerberos01.example.net@EXAMPLE.NET
 2  2  host/kerberos01.example.net@EXAMPLE.NET
ktutil: rkt /root/kerberos02.keytab
ktutil: l
slot KVNO Principal
-----
 1  2  host/kerberos01.example.net@EXAMPLE.NET
 2  2  host/kerberos01.example.net@EXAMPLE.NET
 3  2  host/kerberos02.example.net@EXAMPLE.NET
 4  2  host/kerberos02.example.net@EXAMPLE.NET
```

Listing 14.35 Inhalt der »keytab«-Datei

Wie Sie in dem Beispiel sehen, können Sie beliebig viele *keytab*-Dateien laden und sich den Inhalt aller Dateien gemeinsam anzeigen lassen. Wenn Sie jetzt noch alle Schlüssel in einer Datei zusammenfassen wollen, können Sie alle Schlüssel aus allen geladenen Dateien mit dem Kommando `wkt alle.keytab` in einer Datei speichern. Kopieren Sie die *keytab*-Datei für die Slave-KDCs auf die entsprechenden Hosts ins Verzeichnis */etc*, und nennen Sie die Datei dort *krb5.keytab*.

14.9.3 Konfiguration des KDC-Slaves

Bevor Sie die eigentliche Replikation starten können, richten Sie als Erstes den KDC-Slave ein. Dazu installieren Sie auf dem KDC-Slave für Debian und Ubuntu die Pakete `krb5-config`, `krb5-kdc`, `krb5-user`, `libpam-krb5` und `krb5-kdc-ldap`. Letzteres müssen Sie nur installieren, wenn Sie den Kerberos-Server später in den LDAP einbinden wollen. Auf keinen Fall dürfen Sie das Paket `krb5-admin-server` installieren, da dieses Paket nur auf dem KDC-Master installiert werden darf. Bei openSUSE installieren Sie, wie schon beim KDC-Master, die Pakete `krb5-server` und `krb5-client`.

**Den KDC zu diesem Zeitpunkt noch nicht starten**

Starten Sie den KDC-Dienst zu diesem Zeitpunkt noch nicht auf dem KDC-Slave, denn erst müssen noch ein paar Dateien vom KDC-Master auf den KDC-Slave kopiert werden.



Auf dem KDC-Slave dürfen Sie nur den Dienst `krb5kdc` im Runlevel-Editor des YaST aktivieren. Der Dienst `kadmind` darf auch hier nur auf dem KDC-Master laufen. Bei CentOS benötigen Sie wieder die Pakete `krb5-server`, `krb5-workstation` und für die LDAP-Anbindung das Paket `krb5-server-ldap`. Auch aktivieren Sie nur den KDC-Dienst. Kopieren Sie im Anschluss die folgenden Dateien des KDC-Masters auf den KDC-Slave:

- ▶ die Datei `kdc.conf` für die Serverkonfiguration
- ▶ die Datei `kadm5.acl`, um die Zugriffsrechte gleichzusetzen
- ▶ die Schlüsseldatei. Diese hat entweder den Namen `stash` oder `.k5.EXAMPLE.NET`, je nachdem, wie Sie die Datenbank erzeugt haben. In dieser Datei befindet sich das Master-Passwort, ohne das der Server die Datenbank nicht entschlüsseln kann.

Die Pfade für die letzten drei Dateien sind wieder abhängig von der Distribution, mit der Sie arbeiten. Bei Debian und Ubuntu finden Sie diese im Verzeichnis `/etc/krb5kdc`, bei openSUSE im Verzeichnis `/var/lib/kerberos/krb5kdc` und bei CentOS im Verzeichnis `/var/kerberos/krb5kdc`.

14.9.4 Replikation des KDC-Masters auf den KDC-Slave

In vielen Anleitungen, die im Internet kursieren, wird nun die Datenbank auf dem Slave erst erstellt und dann überschrieben. Das ist aber nicht notwendig und funktioniert in den meisten Fällen auch nicht. Durch das Kopieren der Dateien vom KDC-Master auf den KDC-Slave ist die Vorbereitung des KDC-Slaves abgeschlossen; es muss keine Datenbankdatei mit den Principals vorhanden sein, um eine erfolgreiche Replikation durchführen zu können.

Damit der KDC-Slave auch die Principals vom KDC-Master annimmt, erzeugen Sie jetzt noch die Datei `kpropd.acl` auf dem Slave. Das System sucht die Datei bei Debian und Ubuntu in dem Verzeichnis `/etc/krb5kdc`, bei openSUSE im Verzeichnis `/var/lib/kerberos/krb5dc` und bei CentOS im Verzeichnis `/var/kerberos/krb5kdc`. Den Inhalt der Datei sehen Sie in Listing 14.36:

```
host/kerberos01.example.net@EXAMPLE.NET
```

Listing 14.36 Inhalt der Datei »kpropd.acl«

Leider ist das noch nicht alles. Auf dem Slave benötigen Sie noch den zusätzlichen Dienst `kptopd`, den Sie noch installieren und konfigurieren müssen. Dieser Dienst nimmt die Replikationen des KDC-Masters an und schreibt sie in die Datenbank des KDC-Slaves. Bei diesem Dienst handelt es sich um den `kpropd`. Auch hier ist es wichtig, dass Sie den richtigen Weg für Ihre Distribution wählen.

kpropd bei openSUSE und CentOS

Der *kpropd* läuft nicht als eigenständiger Daemon, sondern wird über den *xinetd* gestartet. Es werden keine weiteren Pakete benötigt, da der *kpropd* Bestandteil des KDC-Pakets ist. Installieren Sie zunächst auf dem Slave den *xinetd*. Nachdem Sie den *xinetd* installiert haben, erzeugen Sie jetzt noch eine Konfigurationsdatei für den *kpropd* im Verzeichnis */etc/xinetd.d*. In Listing 14.37 sehen Sie den Inhalt der Datei *kprop*:

```
service kprop
{
socket_type = stream
wait = no
user = root
server = /usr/sbin/kpropd
only_from = 192.168.56.56
disable = no
}
```

Listing 14.37 Konfigurationsdatei für den »kpropd«

Die IP-Adresse, die in der Datei angegeben ist, ist die IP-Adresse des KDC-Masters, denn nur von dieser IP-Adresse sollen Daten angenommen werden. Der *kpropd* lauscht auf dem TCP-Port 754. Damit der Dienst auch dem Port zugeordnet werden kann, muss ein entsprechender Eintrag in der Datei */etc/services* vorhanden sein. In manchen Distributionen existiert der Eintrag nicht ergänzen Sie den Eintrag dann so, wie in Listing 14.38 zu sehen ist. Anschließend können Sie den *xinetd* neu starten. Mit dem Kommando `ss -tln` wird Ihnen jetzt der TCP-Port 754 als aktiv angezeigt.

```
kprop          754/tcp          # Kerberos propagation
```

Listing 14.38 Eintrag für den »kpropd« in der Datei »/etc/services«

Der kpropd bei Debian und Ubuntu

Auf Debian- bzw. Ubuntu-Systemen installieren Sie das Paket *krb5-kpropd*. Der Dienst selbst wird über den *systemd* gestartet und benötigt den *xinetd* nicht mehr. Auch um den Eintrag in der */etc/services* brauchen Sie sich nicht zu kümmern: Er ist bereits vorhanden. Schreiben Sie im nächsten Schritt auf dem KDC-Master die aktuelle Datenbank in eine Dump-Datei. Diesen Vorgang führen Sie mit dem Kommando `kdb5_util dump /root/slave-repl` durch. Jetzt kommt der spannende Moment, in dem die Datenbank vom KDC-Master zum ersten Mal auf den KDC-Slave propagiert wird. In Listing 14.39 sehen Sie diesen Vorgang:

```
root@kerberos01:~# kdb5_util dump slave-repl
root@kerberos01:~# kprop -d -f slave-repl kerberos02.example.net
4806 bytes sent.
Datenbankverbreitung auf kerberos02.example.net: ERFOLGREICH
```

Listing 14.39 Propagierung der Datenbank

Die Option `-d` schaltet den Debug-Modus für das Kommando `kprop` ein. Anschließend können Sie mit `kdb5_util dump` den Inhalt auf dem KDC-Slave überprüfen. Damit ist die Replikation abgeschlossen, und jetzt können Sie den Kerberos auf dem KDC-Server starten. Sie können auch mit dem Programm `kadmin` auf die Datenbank lesend zugreifen, eine Änderung auf dem Slave ist aber nicht möglich.



Während der Einrichtung von Kerberos erhielten wir die eine oder andere Fehlermeldung. Eine gute Hilfe, um diese Fehler zu beheben, haben wir immer wieder auf der Webseite <http://research.imb.uq.edu.au/~l.rathbone/ldap/kerberos.shtml> gefunden.

Leider wird die Datenbank nicht automatisch vom KDC-Master auf den KDC-Slave propagiert. Diesen Vorgang können Sie aber über ein Shell-Skript in Zusammenarbeit mit einem Cronjob realisieren. In Listing 14.40 sehen Sie ein Beispiel für ein Shell-Skript zur Replikation:

```
#!/bin/bash
/usr/sbin/kdb5_util dump /root/slave-repl
/usr/sbin/kprop -f /root/slave-repl kerberos02.example.net > /dev/null
```

Listing 14.40 Skript für die Replikation



Wie Sie in dem Skript sehen, wird die Option `-d` hier nicht mehr verwendet, da ein Debugging an dieser Stelle nicht mehr notwendig und auch nicht sinnvoll ist.

Da ein Abgleich der beiden Server nicht zeitgleich passiert, kann es vorkommen, dass ein Benutzer, den Sie neu angelegt haben, sich nicht sofort anmelden kann. Sie können die Replikation aber über das Skript von Hand anstoßen. Geänderte Passwörter sind nicht das Problem. Dafür haben Sie ja im DNS den entsprechenden SRV-Eintrag erstellt, der bewirkt, dass bei einer fehlerhaften Authentifizierung immer der KDC-Master mit dem aktuellen Passwort gefunden wird. Ändert ein Benutzer sein Passwort, wird das an den Master geschickt. Meldet der Benutzer sich sofort ab und wieder an und verwendet er dabei sein neues Passwort und landet er mit seiner Anmeldung wieder beim Slave-Server, so kennt dieser das Passwort eventuell noch nicht, da noch keine Replikation stattgefunden hat. In so einem Fall leitet der Slave die Authentifizierung an den Master weiter, der das neue Passwort bereits kennt. Der Master führt die Authentifizierung durch und der Benutzer wird angemeldet.

14.10 Kerberos-Policies

Um die Richtlinien der Passwörter hinsichtlich der Passwörterlänge und Passwortkomplexität zu ändern, gibt es in Kerberos die *Policies*. Mit diesen Policies können Sie unterschiedliche Passwortrichtlinien für verschiedene Benutzergruppen erstellen. Die Policies werden mit dem `kadmin` erzeugt. Für die Policies gibt es die folgenden Parameter:

► `maxlife`

Mit diesem Parameter legen Sie die maximale Gültigkeit des Passworts fest.

- ▶ `minlife`
Mit diesem Parameter legen Sie die minimale Gültigkeit des Passworts fest.
- ▶ `minlength`
Mit diesem Parameter legen Sie die minimale Länge des Passworts fest.
- ▶ `minclasses`
Mit diesem Parameter legen Sie die Komplexität des Passworts fest. Hier legen Sie die Anzahl der unterschiedlichen Klassen der Sonderzeichen fest, die ein Passwort enthalten muss.
Die folgenden Klassen werden von Kerberos verwendet:
 - Kleinbuchstaben
 - Großbuchstaben
 - Zahlen
 - Punkte
 - andere Sonderzeichen
- ▶ `history`
Hiermit legen Sie fest, wie viele alte Passwörter im Principal gespeichert werden.
Diesen Parameter können Sie nicht verwenden, wenn Sie Kerberos zusammen mit LDAP für die Benutzerverwaltung einsetzen wollen.
- ▶ `maxfailure`
Mit diesem Parameter legen Sie die maximale Anzahl an fehlerhaften Anmeldungen fest, die vorkommen dürfen, bis der Benutzer gesperrt wird.
- ▶ `failurecountinterval`
Dieser Parameter legt fest, wie lange die Zeit zwischen zwei fehlerhaften Anmeldungen sein darf. Ist die Zeit abgelaufen, wird der Zähler für die falschen Anmeldungen zurückgesetzt werden. Tragen Sie hier einen Wert von 0 ein, würde der Zähler nie zurückgesetzt. Ist die eingestellte Zeit abgelaufen, wird der Zähler `maxfailure` zurückgesetzt.
- ▶ `lockoutduration`
Legt fest, wie lange ein Konto gesperrt wird, wenn der Wert von `maxfailure` erreicht wurde. Ein Wert von 0 sperrt das Konto für immer oder so lange, bis Sie das Konto mit dem Kommando `kadmin: modify_principal -unlock <Konto>` wieder freischalten.

In Listing 14.41 sehen Sie, wie Policies erstellt werden. Am Ende des Listings werden dann alle gerade erstellten Policies aufgelistet.

```
root@kerbero01:~# kadmin
Authenticating as principal root/admin@EXAMPLE.NET with password.
Password for root/admin@EXAMPLE.NET:
```

```
kadmin: add_policy -minlength 8 -minclasses 3 admin
```



```
kadmin: add_policy -minlength 8 -minlife "2 days" -maxlife "90 days" \  
        -minclasses 3 user  
kadmin: list_policies  
admin  
user
```

Listing 14.41 Erstellen von Policies

In den Beispielen werden für die Principalgruppen `admin` und `user` eigene Policies erstellt. In Listing 14.42 ist dargestellt, wie Sie einen neuen Principal mit einer Policy anlegen können:

```
kadmin: addprinc -policy user ptau  
Geben Sie das Passwort für Principal »ptau@EXAMPLE.NET« ein.:  
Geben Sie das Passwort für Principal »ptau@EXAMPLE.NET« erneut ein.:  
add_principal: Das Passwort ist zu kurz. while creating "ptau@EXAMPLE.NET".
```

```
kadmin: addprinc -policy user ptau  
Geben Sie das Passwort für Principal »ptau@EXAMPLE.NET« ein.:  
Geben Sie das Passwort für Principal »ptau@EXAMPLE.NET« erneut ein.:  
Principal "ptau@EXAMPLE.NET" created.
```

Listing 14.42 Anwenden der Policies

Wie Sie sehen, wird hier dem neuen Principal `ptau` eine Policy zugewiesen. Das hier gewählte Passwort geheim entspricht nicht den Richtlinien, deshalb wird der Benutzer im ersten Versuch auch nicht angelegt. Erst im zweiten Versuch entspricht das Passwort der Policy.

Natürlich sollten Sie Ihre Konfiguration immer gut dokumentieren, so auch die verschiedenen Einstellungen der Policies. Aber selbstverständlich können Sie sich die Einstellungen Ihrer Policies wie in Listing 14.43 auch auflisten lassen:

```
kadmin: get_policy user  
Richtlinie: »user«  
Maximum password life: 90 days 00:00:00  
Minimum password life: 2 days 00:00:00  
minimale Passwortlänge: 8  
minimale Anzahl von Passwortzeichenklassen: 3  
Anzahl aufbewahrter alter Schlüssel: 1  
maximale Anzahl falscher Passwordeingaben vor dem Sperren: 0  
Rücksetzintervall für zu viele falsch eingegebene Passwörter: 0 days 00:00:00  
Passwortsperredauer: 0 days 00:00:00
```

Listing 14.43 Auflisten der Einstellungen

Was ist aber mit dem Benutzer `root`? Der existiert bereits und gehört zur Gruppe `admin`, hat aber bis jetzt noch keine Policy zugewiesen bekommen. Jetzt soll die zuvor erstellte Policy `admin` dem Benutzer `root` zugewiesen werden. Wie Sie das machen, sehen Sie in Listing 14.44:


```
kadmin: modify_principal -policy admin root/admin
Principal »root/admin@EXAMPLE.NET« wurde geändert.
```

Listing 14.44 Zuweisung einer Policy für bestehende Principals

Denken Sie daran, die Datenbank zu replizieren, nachdem Sie die Policies eingerichtet haben. Jetzt haben Sie eine ausfallsicheren Kerberos-Konfiguration. Diese arbeitet aber immer noch nicht mit einer zentralen Benutzerverwaltung zusammen, denn dazu benötigen Sie einen LDAP-Server. Wie Sie beide Dienste zusammenbringen, sehen Sie im nächsten Abschnitt.

14.11 Kerberos in LDAP einbinden

Wenn Sie bis zu diesem Zeitpunkt nur Kerberos eingerichtet haben und alle Benutzer lokal auf den Systemen verwalten, ist der Verwaltungsaufwand hoch, denn immer, wenn Sie ein neues Benutzerkonto benötigen, ist es notwendig, den Benutzer auf allen Hosts lokal anzulegen; das Passwort wird schon zentral im Kerberos verwaltet.

Wir nutzen den OpenLDAP 2.6

Bei unserem Beispiel verwenden wir den OpenLDAP in der Version 2.6. Da zum Zeitpunkt der Erstellung dieses Kapitels noch keine Distribution die neue Version bereitgestellt hat, verwenden wir hier die Pakete der Firma Symas. Die Einrichtung finden Sie in Kapitel 17, »LDAP«. Auch gehen wir hier nur noch auf die dynamische Konfiguration über das Backend *cn=config* ein. Denn nur so lässt sich die Konfiguration später auch auf mehrere Provider replizieren.

Nutzen Sie bereits einen LDAP-Server und wollen Sie jetzt mit Kerberos die Anmeldesicherheit erhöhen, so können Sie Kerberos und LDAP auch weiterhin getrennt verwalten. Aber dann haben Sie zwei Datenbanken: zum einen die LDAP-Datenbank mit allen Benutzereigenschaften und zum anderen die Kerberos-Datenbank mit den Principals und den dazugehörigen Passwörtern.

LDAP ist aber in der Lage, die benötigten Informationen für Kerberos zu speichern und Ihren Kerberos-Servern im Netz bereitzustellen. Dann speichern Sie alle benötigten Kontoinformationen der Benutzer in einem LDAP-Objekt. Der Kerberos-Server greift dann zur Authentifizierung sowohl für die Passwörter als auch für die Principals direkt auf den LDAP-Server zu. Ein weiterer Vorteil dieser Konfiguration ist der, dass Sie keine Replikation der Kerberos-Server benötigen: Alle Kerberos-Server greifen immer auf dieselbe Datenbank zu und haben daher immer dieselben und aktuellen Informationen. Eine Besonderheit bleibt aber bestehen: Auch bei dieser Art der Konfiguration haben Sie immer nur einen Admin-Server für den Kerberos.





Hier gehen wir davon aus, dass Sie bereits einen wie in Kapitel 17, »LDAP«, eingerichteten LDAP-Server in Ihrem Netz nutzen. Wenn das nicht der Fall ist, ist es sinnvoll, dass Sie als Erstes einen LDAP-Server einrichten, da hier nur noch die Änderungen am LDAP-Server angesprochen werden.

Wir besprechen im LDAP-Kapitel auch die beiden Möglichkeiten der Konfiguration des LDAP-Servers, einmal die statische Konfiguration über die ASCII-Textdatei */etc/ldap/slapd.conf* und dann auch die dynamische Konfiguration über das Backend *cn=config*. Damit Sie die Möglichkeit haben, auch bei der Einbindung des Kerberos in dem LDAP beide Wege zu gehen, werden wir auch hier beide Wege beschreiben.

An dieser Stelle noch einmal der Hinweis: Wir verwenden nur noch Debian und Ubuntu für OpenLDAP, da OpenLDAP von CentOS und zum Teil auch schon von openSUSE nicht mehr unterstützt wird. Mehr dazu finden Sie im LDAP-Kapitel.

14.11.1 Konfiguration des LDAP-Servers



Stellen Sie sicher, dass auf allen Kerberos-Servern das Paket *krb5-kdc-ldap* installiert ist, da ohne dieses Paket die Einbindung von Kerberos in den LDAP-Server nicht möglich ist.

Das Schemas in die dynamischen Konfiguration eintragen

Das Kerberos-Schema, das Sie für den OpenLDAP benötigen, befindet sich nicht in den OpenLDAP-Paketen, sondern in dem Paket *krb5-kdc-ldap*. Abhängig von der Distribution finden Sie nach der Installation des Pakets nur die Datei */usr/share/doc/krb5-kdc-ldap/kerberos.schema.gz*, oder die Dateien */usr/share/doc/krb5-kdc-ldap/kerberos.openldap.ldif.gz* und */usr/share/doc/krb5-kdc-ldap/kerberos.schema.gz*. Wenn Sie nur die Datei */usr/share/doc/krb5-kdc-ldap/kerberos.schema.gz* finden und die dynamische Konfiguration nutzen, ist es notwendig, dass Sie die Datei erst in eine LDIF-Datei umwandeln. Der nächste Abschnitt beschreibt die Umwandlung der Schema-Datei in eine LDIF-Datei. Haben Sie bereits eine LDIF-Datei können Sie den nächsten Abschnitt überspringen.

Umwandeln der Schema-Datei

Kopieren Sie die Datei in das Verzeichnis */etc/ldap/schema* Ihres LDAP-Servers, und entpacken Sie die Datei dort. Im Anschluss erstellen Sie ein Verzeichnis für die Umwandlung. Wir verwenden hier das Verzeichnis */root/make-ldif*. Erstellen Sie jetzt eine Datei *schema-convert.conf*. In diese Datei schreiben Sie nur die eine Zeile aus Listing 14.45:

```
include /etc/ldap/schema/kerberos.schema
```

Listing 14.45 Datei zur Umwandlung des Schemas

Die Datei darf nicht mehr in dem vorher angelegten Verzeichnis liegen. Das Verzeichnis muss leer bleiben.

Anschließend wandeln Sie die Datei in eine dynamische Konfiguration mit `slaptest` um. In Listing 14.46 sehen Sie das Kommando:

```
root@ldap01:~# slaptest -f schema-convert.conf -F /root/make-ldif
config file testing succeeded
```

Listing 14.46 Die Umwandlung

Das Kommando erzeugt eine sehr einfache dynamische Konfiguration, die lediglich aus den Informationen für den Kerberos besteht. Wichtig ist nur die Datei `/make-ldif/cn=config/cn=schema/cn={O}kerberos.ldif`. Diese Datei benötigt aber noch zwei Änderungen. Öffnen Sie die Datei mit einem Editor, und passen Sie als Erstes den `dn` wie in Listing 14.47 an. Im Listing sehen Sie sowohl den originalen Eintrag wie auch den geänderten Eintrag:

```
## Start Original ##
dn: cn=0kerberos
objectClass: olcSchemaConfig
cn: 0kerberos
## Ende Original ##
## Start Aenderung ##
dn: cn=kerberos,cn=schema,cn=config
changeType: add
objectClass: olcSchemaConfig
cn: kerberos
## Ende Aenderung ##
```

Listing 14.47 Anpassen des dn

Aus derselben Datei entfernen Sie die Zeilen, die Sie in Listing 14.48 sehen, am Ende der Datei:

```
structuralObjectClass: olcSchemaConfig
entryUUID: 035ebf5c-d987-103a-90c4-1b52df92132b
creatorsName: cn=config
createTimestamp: 20201223162402Z
entryCSN: 20221223162402.455629Z#000000#000#000000
modifiersName: cn=config
modifyTimestamp: 20221223162402Z
```

Listing 14.48 Diese Zeilen entfernen

Speichern Sie die Datei, und kopieren Sie sie nach `/etc/ldap/schema/kerberos.ldif`. Anschließend spielen Sie die Datei wie in Listing 14.49 in die LDAP-Konfiguration ein:

```
root@ldap01:~# ldapmodify -Y EXTERNAL -H ldapi:/// \
-f /etc/ldap/schema/kerberos.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
```

```
SASL SSF: 0
adding new entry "cn=kerberos,cn=schema,cn=config"
```

Listing 14.49 Einspielen der LDIF-Datei

Damit haben Sie das zusätzlich benötigte Schema in die dynamische Konfiguration eingebunden. Nutzen Sie mehrere LDAP-Server, stellen Sie sicher, dass das Schema auf allen LDAP-Servern eingebunden ist.

Einspielen des Schemas mit einer vorhandenen LDIF-Datei

Wenn Sie die Datei `/usr/share/doc/krb5-kdc-ldap/kerberos.openldap.ldif.gz` auf Ihrem Kerberos-Server finden, dann kopieren Sie die Datei in das Schemaverzeichnis Ihres LDAP-Servers.



Da wir hier im Buch die OpenLDAP-Pakete der Firma Symas nutzen, wäre das der Pfad `/opt/symas/etc/openldap/schema`.

Entpacken Sie die Datei auf dem LDAP-Server, und spielen Sie das Schema mit dem Kommando `ldapadd -Y EXTERNAL -H ldapi:/// -f kerberos.openldap.ldif` ein. Wenn Sie mehrere LDAP-Server eingerichtet haben, stellen Sie sicher, dass das Schema auf allen LDAP-Servern installiert wird.

Anlegen der benötigten Objekte

Da der Kerberos-Server die Objekte für die Principals verwalten soll, benötigen Sie zwei Objekte, die dafür die Rechte erhalten und mit denen sich der Kerberos am LDAP authentifizieren kann. Für die Verwaltung der Objekte legen Sie am besten eine neue *Organizational Unit* an. Innerhalb dieser *ou* werden zwei weitere Objekte angelegt: zum einen ein Objekt für den `kdc` und zum anderen ein Objekt für `kadmin`.

Diese Objekte werden nicht unbedingt benötigt. Sie können auch für die Verwaltung des Kerberos-Servers im LDAP-Baum immer den *rootdn* des LDAP-Servers verwenden. Aus Sicherheitsgründen ist es aber angebracht, mit zwei zusätzlichen Objekten zu arbeiten, da diese beiden Objekte nur Zugriff auf den Kerberos-Teil Ihres LDAP-Baums benötigen. In Listing 14.50 sehen Sie die entsprechenden LDIF-Einträge:

```
dn: ou=kerberos-adm,dc=example,dc=net
ou: kerberos-adm
objectClass: organizationalUnit
objectClass: top

dn: cn=kdc,ou=kerberos-adm,dc=example,dc=net
cn: kdc
objectClass: organizationalRole
objectClass: simpleSecurityObject
userPassword: {ARGON2}....
```

```
dn: cn=kadmin,ou=kerberos-adm,dc=example,dc=net
cn: kadmin
objectClass: organizationalRole
objectClass: simpleSecurityObject
userPassword: {ARGON2}....
```

Listing 14.50 LDIF-Einträge für die ersten Objekte

Die Passwörter können Sie mit dem Kommando `echo -n 'mein-passwort' | argon2 'saltwert' -e` erzeugen und dann in die LDIF-Datei eintragen. Die *Organizational Unit* wird später nicht für die eigentliche Datenbank verwendet, sondern dient nur dazu, die beiden administrativen Objekte anzulegen. Der Container für die Kerberos-Principals wird während der Initialisierung der Datenbank angelegt. Wichtig ist, dass Sie jetzt schon mit ACLs im LDAP-Verzeichnis dafür sorgen, dass die beiden administrativen Objekte später auch schreibenden Zugriff auf die Kerberos-Daten haben.



Anpassen der Limits

Denken Sie daran, dass eventuell die maximal angezeigten Suchlimits auf 500 begrenzt sind. Kerberos muss aber alle Objekte durchsuchen können. Deshalb heben Sie auf jeden Fall für beide Objekte diese Limits auf, so wie Sie das in Listing 14.51 sehen.



14

Anpassen der ACLs

Die beiden Objekte `cn=kdc` und `cn=kadmin` benötigen im LDAP-Verzeichnis Schreibrechte an den Benutzerobjekten und den Kerberos-Objekten. In Listing 14.51 sehen Sie die ACLs, die sicherstellen, dass die beiden Objekte die benötigten Rechte erhalten. Die Einrichtung der ACLs wird mittels einer LDIF-Datei durchgeführt. In Listing 14.51 sehen Sie die LDIF-Datei, mit der Sie die ACLs anpassen können:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
delete: olcAccess
olcAccess: {0}
-
add: olcAccess
olcAccess: {0}to *
  by dn.exact="uid=sssd-user,ou=users,dc=example,dc=net" read
  by dn.exact="uid=repl-user,ou=users,dc=example,dc=net" read
  by dn.exact="uid=ldap-admin,ou=users,dc=example,dc=net" write
  by dn.exact="cn=kdc,ou=kerberos-adm,dc=example,dc=net" write
  by dn.exact="cn=kadmin,ou=kerberos-adm,dc=example,dc=net" write
```

```

by dn.exact="krbPrincipalName=K/M@EXAMPLE.NET,cn=EXAMPLE.NET,\
   cn=kerberos,dc=example,dc=net" write
by * break
-
add: olcLimits
olcLimits: {1} dn.exact="cn=kdc,ou=kerberos-adm,dc=example,dc=net"
   time=unlimited
   size=unlimited
-
add: olcLimits
olcLimits: {2} dn.exact="cn=kadmin,ou=kerberos-adm,dc=example,dc=net"
   time=unlimited
   size=unlimited

```

Listing 14.51 LDIF-Datei für die ACLs der dynamischen Konfiguration



Um die ACLs an dieser Stelle nicht zu kompliziert werden zu lassen, haben wir die bestehende ACL auf alle Objekte hier nur erweitert. Wenn Sie gezielt vorgehen wollen, dann setzen Sie die ACLs nur auf die Container, in denen sich später Benutzer befinden, die Kerberos-Attribute erhalten sollen. Sollten Sie bereits einen LDAP-Server nutzen, achten Sie auf die korrekte Nummerierung der Datenbanken.

Um die Performance des LDAP-Servers zu verbessern, sollten Sie immer die beiden Attribute `krbPrincipalName` und `krbPwdPolicyReference` indizieren. In Listing 14.52 sehen Sie die LDIF-Datei, um die Index-Einträge in die Konfiguration zu übernehmen:

```

dn: olcDatabase={2}mdb,cn=config
changetype: modify
add: olcDbIndex
olcDbIndex: krbPrincipalName,krbPwdPolicyReference eq

```

Listing 14.52 LDIF-Datei für die Index-Einträge

Umstellung des Kerberos-Servers

Im ersten Schritt erzeugen Sie ein Backup der Kerberos-Datenbank. Dieses Backup benötigen Sie später auch noch, um die Daten in den LDAP-Baum zu übernehmen. Stoppen Sie den *KDC* und den *kadmin-server* auf Ihrem Kerberos-Master-Server. Dann erstellen Sie das Backup mit dem Kommando `kdb5_util dump /root/example.net.backup`.



Alte Datenbank löschen

Vergessen Sie nicht, die alte Datenbank mit den Principals zu löschen. Diese Datenbank wird nach der Umstellung nicht mehr benötigt. Am einfachsten löschen Sie die Datenbank mit dem Kommando `kdb5_util destroy`.

In diesem Backup finden Sie alle Principals und alle Richtlinien, die Sie bis zu diesem Zeitpunkt im Kerberos erzeugt haben. Passen Sie auf Ihrem Kerberos-Server die Datei `/etc/krb5kdc/kdc.conf` so an, dass der Kerberos-Server die Datenbank in Ihrem LDAP-Server sucht und nicht mehr in der lokalen Datenbank (siehe Listing 14.53).

```
[kdcdefaults]
    kdc_ports = 750,88

[realms]
    EXAMPLE.NET = {
        admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
        acl_file = /etc/krb5kdc/kadm5.acl
        key_stash_file = /etc/krb5kdc/stash
        kdc_ports = 750,88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = des3-hmac-sha1
        #supported_encetypes = aes256-cts:normal aes128-cts:normal
        default_principal_flags = +preauth
        database_module = ldapconf
    }

[logging]
    kdc = FILE:/var/log/krb5/krb5kdc.log
    admin_server = FILE:/var/log/krb5/kadmind.log

[dbmodules]
    ldapconf = {
        db_library = kldap
        ldap_kerberos_container_dn = "cn=kerberos,dc=example,dc=net"
        ldap_kdc_dn = "cn=kdc,ou=kerberos-adm,dc=example,dc=net"
        ldap_kadmind_dn = "cn=kadmin,ou=kerberos-adm,dc=example,dc=net"
        ldap_service_password_file = "/etc/krb5kdc/service.keyfile"
        ldap_servers = "ldaps://ldap01.example.net"
        ldap_conns_per_server = 5
    }
```

Listing 14.53 Die neue Datei »kdc.conf«

Die neuen Parameter haben die folgenden Bedeutungen:

- ▶ `database_module = ldapconf`
Damit die Parameter für den Zugriff auf den LDAP-Server auch für jeden Realm anders gesetzt werden können, wird hier der neue Abschnitt für die Modulkonfiguration gesetzt. Den Namen können Sie frei wählen.

- ▶ [dbmodules]
Hier beginnt der neue Abschnitt für die Konfiguration des Zugriffs auf den LDAP-Server.
- ▶ ldapconf = {
An dieser Stelle beginnt die Konfiguration für den ausgewählten Realm. Hier verwenden Sie den Namen, den Sie zuvor festgelegt haben.
- ▶ db_library = kldap
Alle anderen Backends als das Standard-Backend (die lokale Datenbank) werden beim MIT-Kerberos über Module geladen. Für die Verwendung von LDAP als Backend ist das Modul kldap verantwortlich, das an dieser Stelle geladen wird.
- ▶ ldap_kerberos_container_dn = "cn=kerberos,dc=example,dc=net"
Dieser Parameter legt den Container im LDAP-Baum fest, in dem die Kerberos-Objekte verwaltet werden.
- ▶ ldap_kdc_dn = "cn=kdc,ou=kerberos-adm,dc=example,dc=net"
Hier geben Sie das Objekt des KDC an.
- ▶ ldap_kadmin_dn = "cn=kadmin,ou=kerberos-adm,dc=example,dc=net"
Hier geben Sie das Objekt für den *kadmin* an.
- ▶ ldap_service_password_file = /etc/krb5kdc/service.keyfile"
An dieser Stelle geben Sie die Datei an, in der später das Passwort für den *LDAP-bind* abgelegt wird.
- ▶ ldap_servers = "ldaps://k1-stat.example.net"
Hier können Sie, durch Leerzeichen voneinander getrennt, Ihre LDAP-Server angeben. Damit die Verbindung auch verschlüsselt ist, wird hier ldaps verwendet.
- ▶ ldap_conns_per_server = 5
Die Anzahl der Verbindungen, die der Kerberos-Server zum LDAP-Server geöffnet halten soll. Der Wert von 5 ist dabei der Standardwert. Nur wenn die Authentifizierung über mehrere LDAP-Server geht, muss dieser Wert erhöht werden.



Die Datei `/etc/ldap/ldap.conf` anpassen

Vergessen Sie nicht, auf dem Kerberos-Server die Datei `/etc/ldap/ldap.conf` so anzupassen, wie Sie es in Listing 14.54 sehen. Kopieren Sie zusätzlich das root-Zertifikat an die entsprechende Stelle. An dieser Stelle ist der Pfad zur Datei `ldap.conf` immer `/etc/ldap/ldap.conf`, da auf dem Kerberos-Server nicht die Symas-Pakete installiert wurden.

```
BASE    dc=example,dc=net
URI     ldaps://ldap01.example.net

TLS_CACERT    /etc/ssl/zertifikate/demoCA/cacert.pem
```

Listing 14.54 Anpassen der Datei »ldap.conf«

Im Anschluss daran können Sie die neue LDAP-basierte Datenbank mit dem Kommando `kdb5_ldap_utils` initialisieren. In Listing 14.55 sehen Sie den Initialisierungsvorgang:

```
root@kerberos01:~# kdb5_ldap_util create -D cn=admin,dc=example,dc=net \
-r EXAMPLE.NET -s -sscope sub
Passwort für »cn=admin,dc=example,dc=net«:
Datenbank für Realm »EXAMPLE.NET« wird initialisiert
Sie werden nach dem Master-Passwort der Datenbank gefragt.
Es ist wichtig, dass Sie dieses Passwort NICHT VERGESSEN.
Enter KDC database master key:
Re-enter KDC database master key to verify:
```

Listing 14.55 Initialisierung der Kerberos-Datenbank im LDAP-Baum

Bei dem Kommando wird die Option `-s` verwendet, damit der Master-Key der Datenbank in eine `stash`-Datei geschrieben wird. Nach der Initialisierung der Datenbank sehen Sie im LDAP-Baum die Objekte wie in Listing 14.56.

Wahl der Master-Keys

Sie sollten beim Master-Key für die Datenbank unbedingt denselben Schlüssel verwenden wie schon bei der lokalen Datenbank. Dies ist notwendig, damit die Entschlüsselung der lokalen Datenbank im LDAP-Server vorgenommen werden kann.



14

```
root@ldap01:/etc# ldapsearch -x -D uid=ldap-admin,ou=users,dc=example,dc=net\
-W attrs dn -LLL
[...]
dn: ou=kerberos-adm,dc=example,dc=net

dn: cn=kdc,ou=kerberos-adm,dc=example,dc=net

dn: cn=kadmin,ou=kerberos-adm,dc=example,dc=net

dn: cn=kerberos,dc=example,dc=net

dn: cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net

dn: krbPrincipalName=K/M@EXAMPLE.NET,cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net

dn: krbPrincipalName=krbtgt/EXAMPLE.NET@EXAMPLE.NET,cn=EXAMPLE.NET,\
cn=kerberos,dc=example,dc=net

dn: krbPrincipalName=kadmin/admin@EXAMPLE.NET,cn=EXAMPLE.NET,\
cn=kerberos,dc=example,dc=net
```

```
dn: krbPrincipalName=kadmin/k1-stat.example.net@EXAMPLE.NET,\
    cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net
```

```
dn: krbPrincipalName=kiprop/k1-stat.example.net@EXAMPLE.NET,\
    cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net
```

```
dn: krbPrincipalName=kadmin/changepw@EXAMPLE.NET,cn=EXAMPLE.NET,\
    cn=kerberos,dc=example,dc=net
```

```
dn: krbPrincipalName=kadmin/history@EXAMPLE.NET,cn=EXAMPLE.NET,\
    cn=kerberos,dc=example,dc=net
```

Listing 14.56 Kerberos-Objekte im LDAP-Baum

Bei diesen Objekten handelt es sich um dieselben Einträge, die schon bei der Initialisierung des Kerberos ohne LDAP erstellt wurden. Diese Objekte werden vom Kerberos-Server benötigt, Sie sollten diese Objekte daher nicht löschen oder verändern. Wie schon nach der Installation des Kerberos in einer lokalen Datenbank wird jetzt als Erstes das Passwort für den `cn=kdc` und den `cn=kadmin` in der *stash*-Datei abgelegt. In Listing 14.57 können Sie diesen Vorgang nachvollziehen:

```
root@kerberos01:~# kdb5_ldap_util stashesrvpw -D cn=admin,dc=example,dc=net -f \
    /etc/krb5kdc/service.keyfile cn=kdc,ou=kerberos-adm,dc=example,dc=net
Passwort für »cn=admin,dc=example,dc=net«:
Passwort für »cn=kdc,ou=kerberos-adm,dc=example,dc=net«:
Geben Sie das Passwort für »cn=kdc,ou=kerberos-adm,dc=example,dc=net« erneut ein.:
root@kerberos01:~# kdb5_ldap_util stashesrvpw -D cn=admin,dc=example,dc=net -f \
    /etc/krb5kdc/service.keyfile cn=kadmin,ou=kerberos-adm,dc=example,dc=net
Passwort für »cn=admin,dc=example,dc=net«:
Passwort für »cn=kadmin,ou=kerberos-adm,dc=example,dc=net«:
Geben Sie das Passwort für »cn=kadmin,ou=kerberos-adm,dc=example,dc=net« erneut ein.:
```

Listing 14.57 Erstellung der »stash«-Datei

Als Datei für die Passwörter wird hier die in Listing 14.53 angegebene Datei */etc/krb5kdc/service.keyfile* verwendet. Achten Sie darauf, dass in der Datei *service.keyfile* der komplette DN der Objekte steht, so wie es in Listing 14.58 zu sehen ist:

```
cn=kdc,ou=kerberos-adm,dc=example,dc=net#{HEX}67656865696d
cn=kadmin,ou=kerberos-adm,dc=example,dc=net#{HEX}67656865696d
```

Listing 14.58 Inhalt der Datei »service.keyfile«


Jetzt können Sie sich mit dem Kommando `kadmin.local -q listprincs` alle bis jetzt erstellten Principals auflisten lassen. Erst wenn das ohne Fehler geht, sollten Sie mit dem nächsten Schritt fortfahren. Sie sehen bei der Auflistung erst nur die Standard-Principals.

14.11.2 Zurücksichern der alten Datenbank

Im nächsten Schritt kann jetzt die alte Kerberos-Datenbank, die Sie als *dump* gesichert haben, eingespielt werden. Hierzu wird das Kommando `kdb5_util` verwendet. In Listing 14.59 sehen Sie den Vorgang des Imports und das anschließende Auflisten aller Principals:

```
root@kerberos01:~# kdb5_util load -update example.net.backup
root@kerberos01:~# kadmin.local -q listprincs
Authentifizierung als Principal root/admin@EXAMPLE.NET mit Passwort
K/M@EXAMPLE.NET
krbtgt/EXAMPLE.NET@EXAMPLE.NET
kadmin/admin@EXAMPLE.NET
kadmin/k1-stat.example.net@EXAMPLE.NET
kiprop/k1-stat.example.net@EXAMPLE.NET
kadmin/changepw@EXAMPLE.NET
kadmin/history@EXAMPLE.NET
host/k1-stat.example.net@EXAMPLE.NET
host/k2-stat.example.net@EXAMPLE.NET
ldap/k1-stat.example.net@EXAMPLE.NET
ptau@EXAMPLE.NET
root/admin@EXAMPLE.NET
stefan@EXAMPLE.NET
```

Listing 14.59 Importieren der gesicherten Kerberos-Datenbank

Wenn Sie auf einer zweiten Konsole ein `journalctl -f` parallel starten, können Sie die folgenden Schritte beobachten. So können Sie auch erkennen, dass der Kerberos-Server für die Authentifizierung jetzt auf den LDAP-Server zugreift. 

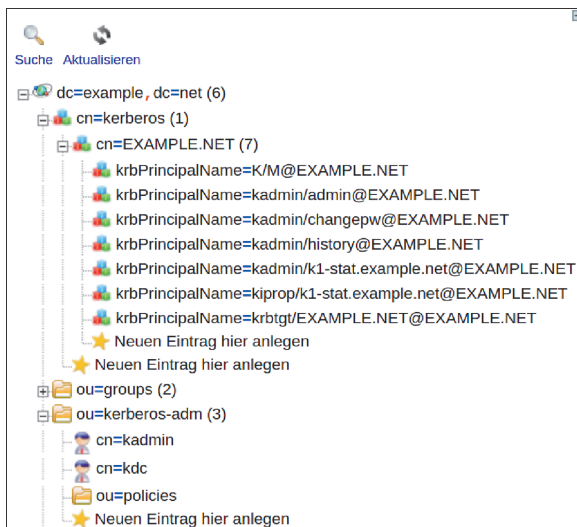


Abbildung 14.1 Kerberos-Objekte im »LAM Pro«

Testen Sie anschließend auf Ihrem LDAP-Server wieder mit `ldapsearch -x -D uid=ldap-admin,ou=users,dc=example,dc=net -W -LLL dn`, ob auch alle Principals im LDAP-Baum angelegt wurden. Wenn das der Fall ist, haben Sie die Datenbank erfolgreich zurückgesichert. Starten Sie jetzt die Dienste für den Kerberos-Server. In Abbildung 14.1 sehen Sie das Ergebnis im *LDAP Account Manager (LAM Pro)*.

Wie Sie in der Abbildung sehen, können Sie in der Pro-Version des LAM auch die Policies verwalten. Selbstverständlich können Sie die Policies weiterhin über die Kommandozeile erstellen und verwalten, aber es gibt auch die Möglichkeit, die Policies als eigene Objekte im LDAP zu verwalten. Wenn Sie den LAM Pro nutzen und bei der Einrichtung das Modul für die Policies eingebunden haben, wurde beim ersten Start gleich der von Ihnen angegebene Container für die Policies angelegt und Sie sehen den zusätzlichen Karteireiter im LAM für die Policies. Der Reiter heißt hier MIT KERBEROS RICHTLINIEN. Wenn Sie dort auf NEUE RICHTLINIE klicken, können Sie alle gewünschten Einstellungen vornehmen. In Abbildung 14.2 sehen Sie die neue Richtlinie für die Gruppe der Administratoren.

The screenshot shows the 'MIT Kerberos Richtlinien' tab in the LAM Pro interface. At the top, there are navigation tabs for 'Benutzer', 'Gruppen', and 'MIT Kerberos Richtlinien'. Below these are buttons for 'Speichern', 'Änderungen zurücksetzen', and 'Löschen'. The main content area is titled 'Admin' and shows a breadcrumb path: 'Suffix policies > kerberos-adm > example > net'. A 'Passwortrichtlinie' (Password Policy) is being configured with the following settings:

Parameter	Value
Name *	Admin
Minimales Passwortalter	2s
Maximales Passwortalter	1m30s
Minimale Passwortlänge	7
Passwort-Historienlänge	10
Maximale Fehlerzahl	4
Sperrdauer	1h
Fehlerzählerintervall	2h
Minimale Anzahl von Zeichenklassen	2
Erlaubte Schlüssel/Salt-Typen	

Abbildung 14.2 Erstellung einer neuen Richtlinie

Die importierten Policies

Selbstverständlich wurden Ihre bereits erstellten Policies mit in den LDAP übernommen. Sie finden sie in der OU `CN=EXAMPLE.NET,cn=kerberos,dc=example,dc=net`. Auch die bestehenden Policies können Sie mit dem LAM bearbeiten.

Jetzt können Sie testen, ob sich die bestehenden LDAP-Benutzer, die schon einen Principal besitzen, auch authentifizieren können. Den Test können Sie mit dem Kommando `kinit` so durchführen, wie Sie es in Listing 14.60 sehen:

```
root@kerberos01:~# kinit stefan
Passwort für stefan@EXAMPLE.NET:
root@k1-stat:~# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: stefan@EXAMPLE.NET
```

```
Valid starting      Expires            Service principal
09.12.2022 17:47:35  10.12.2022 03:47:35  krbtgt/EXAMPLE.NET@EXAMPLE.NET
                erneuern bis 10.12.2022 17:47:33
```

Listing 14.60 Testen der Authentifizierung mit »kinit«

Wollen Sie jetzt einem Principal eine neue Policy zuweisen, fügen Sie das Attribut `krbPwPolicyReference` zu dessen Objekt hinzu. In Listing 14.61 sehen Sie den Principal für den Benutzer `root`. Dort finden Sie auch das zugewiesene Attribut für die Policy:

```
root@ldap01:~# ldapsearch -x -D uid=ldap-admin,ou=users,dc=example,dc=net \
                        -W -LLL krbPrincipalName=root/admin@EXAMPLE.NET
Enter LDAP Password:
dn: krbPrincipalName=root/admin@EXAMPLE.NET,cn=EXAMPLE.NET,cn=kerberos,\
    dc=example,dc=net
krbLastSuccessfulAuth: 20221209152741Z
krbLastFailedAuth: 19700101000000Z
krbLoginFailedCount: 0
krbMaxTicketLife: 36000
krbMaxRenewableAge: 604800
krbTicketFlags: 128
krbPrincipalName: root/admin@EXAMPLE.NET
krbPrincipalExpiration: 19700101000000Z
krbPasswordExpiration: 19700101000000Z
krbPwPolicyReference: cn=admin,cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net
krbPrincipalKey:: MIG2oAMCAQGhAwIBAAIDAgEBowMCAQKgZ8wgZwwVKAHMAWAwIBAKFJMEeg
AwIBEqFABD4gADioHJe/zOZqmn2TMN5YrZWNEZ8jf10K19Vagb6tbJs/Bw4CZu8Ty9AzVy/8mZN5M
qvKoPMGS9yXrCDYzjBEoAcwBaADAgEAoTkwn6ADAgERoTAEhAA/dC8XyypvYFS6cNGXE14ggA7I=
krbLastPwdChange: 20221208154244Z
krbExtraData:: AAKIVJNjcm9vdC9hZG1pbkBFWEFNUEXFLk5FVAA=
krbExtraData:: AAgBAA==
objectClass: krbPrincipal
objectClass: krbPrincipalAux
objectClass: krbTicketPolicyAux
```

Listing 14.61 Principal für den »root«

Es fällt schon auf, dass hier wirklich nur die Kerberos-Informationen gespeichert wurden, mehr war auch in der importierten Datenbank nicht vorhanden. Dasselbe gilt auch für alle anderen Principals. Die eigentliche Benutzerinformation für die Anmeldung wurde ja bis zu dem Punkt in einem lokalen Konto verwaltet. Nur neue Benutzer können Sie schon komplett, inklusive der Kerberos-Information, im LDAP ablegen. Später zeigen wir Ihnen noch, wie Sie bereits im LDAP vorhandene Benutzer um die Kerberos-Informationen erweitern können.

14.11.3 Erstellung der Service-Keys in der Standard-»keytab«-Datei

Bevor Sie hier fortfahren, sorgen Sie dafür, dass für Ihren LDAP-Server ein Principal existiert und dass Sie bereits eine zentrale *keytab*-Datei für den Host erzeugt haben. Die Schritte, die Sie benötigen, um den Principal anzulegen, die zentrale *keytab*-Datei für den LDAP-Server zu erzeugen und sie dann auf den Server zu kopieren, sehen Sie in Listing 14.62:

```
kadmin: addprinc -randkey host/ldap01.example.net
WARNUNG: Für host/ldap01.example.net@EXAMPLE.NET wurde keine Richtlinie angegeben, \
        es wird die Vorgabe »keine Richtlinie« verwendet.
Principal "host/ldap01.example.net@EXAMPLE.NET" created.

kadmin: addprinc -randkey ldap/ldap01.example.net
WARNUNG: Für ldap/ldap01.example.net@EXAMPLE.NET wurde keine Richtlinie angegeben, \
        es wird die Vorgabe »keine Richtlinie« verwendet.
Principal "ldap/ldap01.example.net@EXAMPLE.NET" created.

kadmin: ktadd -k /root/ldap01-krb5.keytab host/ldap01.example.net
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 \
    und Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
    Schlüsseltabelle WRFILE:/root/ldap01-krb5.keytab hinzugefügt.
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 \
    und Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
    Schlüsseltabelle WRFILE:/root/ldap01-krb5.keytab hinzugefügt
kadmin: q
root@k1-stat:~# scp /root/ldap01-krb5.keytab ldap01:/etc/krb5.keytab
```

Listing 14.62 Erstellen des LDAP-Principals

So haben Sie jetzt einen Principal und eine *krb5.keytab* für den eigenständigen LDAP-Server erzeugt. Die Authentifizierung eines Benutzers funktioniert jetzt schon. Damit der LDAP-Server bei der Authentifizierung auch auf den Kerberos-Server zugreifen kann, benötigen Sie noch einen Service-Principal für den LDAP-Server und einen *keytab*-Datei. Wir werden hier eine eigene *keytab*-Datei für den LDAP erstellen. Dadurch sind Sie erheblich flexibler, wenn Sie einmal einen Dienst auf einen anderen Server verschieben wollen. In Listing 14.63 zeigen wir Ihnen, wie Sie die Service-*keytab*-Datei erzeugen:

```

kadmin: ktadd -k /root/ldap01-ldap.keytab ldap/ldap01.example.net
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 \
  und Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
  Schlüsseltabelle WRFILE:/root/ldap01-ldap.keytab hinzugefügt.
Der Eintrag für Principal ldap/ldap01.example.net mit KVNO 2 \
  und Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
  Schlüsseltabelle WRFILE:/root/ldap01-ldap.keytab hinzugefügt
kadmin: q
root@k1-stat:~# scp /root/ldap01-ldap.keytab ldap01:/opt/symas/etc/openldap\
  /ldap.keytab

```

Listing 14.63 Erstellen des Principals für den LDAP-Server

Jetzt haben Sie zwei Schlüsseldateien auf Ihrem LDAP-Server: einmal die `/etc/krb5.keytab` für die Host-Authentifizierung und die `/opt/symas/etc/openldap/ldap.keytab` für die Service-Authentifizierung.

Achten Sie auf die Rechte

Damit der LDAP-Server den Key auch lesen kann, achten Sie unbedingt darauf, dass die Rechte an der `keytab`-Datei richtig gesetzt sind. Die Datei sollte immer dem Benutzer `root` gehören, der dann die Rechte `read` und `write` besitzt. Als Gruppe tragen Sie immer die Gruppe ein, unter der der LDAP-Dienst läuft. Die Gruppe benötigt nur das Recht `read`. Ohne die passenden Berechtigungen kann der LDAP-Server später keine Authentifizierung mithilfe der `keytab`-Datei durchführen.



14

Gerade die benötigten Rechte machen es sinnvoll, die Schlüssel jeweils in einer eigenen Datei abzulegen, denn was machen Sie, wenn Sie mehrere Dienste auf einem Host laufen haben und alle Dienste eigene Benutzer und Gruppen besitzen? Wenn Sie bereits eine Replikation für den LDAP eingerichtet haben, wiederholen Sie die Schritte für alle Consumer, bevor Sie mit dem nächsten Abschnitt weitermachen. Vergessen Sie auch nicht, die Datei `/etc/krb5.conf` auf die Consumer zu kopieren.

14.11.4 Bestehende LDAP-Benutzer um Kerberos-Principal erweitern

Bis zu diesem Zeitpunkt werden die Principals und die Benutzerinformationen noch in getrennten Objekten verwaltet. Jetzt sollen die Benutzerobjekte und die Principals zu einem Objekt zusammengefasst werden. Dazu erweitern Sie den Suchpfad für die Principals des Kerberos-Servers um den LDAP-Container, in dem Sie Ihre Benutzer verwalten. Diese Erweiterung führen Sie wieder mit dem Kommando `kdb5_ldap_util` durch, so wie Sie es in Listing 14.64 sehen:

```

root@kerberos01:~# kdb5_ldap_util modify -D cn=admin,dc=example,dc=net \
-r EXAMPLE.NET -subtrees ou=verwaltung,dc=example,dc=net:\
ou=produktion,dc=example,dc=net:ou=users,dc=example,dc=net
Passwort für »cn=admin,dc=example,dc=net«:

root@kerberos01:~# kdb5_ldap_util view -D cn=admin,dc=example,dc=net -subtrees
Passwort für »cn=admin,dc=example,dc=net«:
    Realm-Name: EXAMPLE.NET
    Teilbaum: ou=verwaltung,dc=example,dc=net
    Teilbaum: ou=produktion,dc=example,dc=net
    Teilbaum: ou=users,dc=example,dc=net
    Suchbereich: SUB

```

Listing 14.64 Erweitern des Suchpfads für Kerberos

Wenn Sie, wie im Beispiel, mehrere OUs angeben wollen, dann trennen Sie die OUs mit einem Doppelpunkt.



Liste wird überschrieben

Wenn Sie später die Liste ändern wollen, achten Sie darauf, dass die Liste immer überschrieben und nicht ergänzt wird. Geben Sie immer alle OUs an, die durchsucht werden sollen.

Nachdem Sie den Suchpfad erweitert haben, können Sie das Ergebnis mit dem Kommando `ldapsearch` überprüfen, wie Sie es in Listing 14.65 sehen:

```

root@ldap01:~# ldapsearch -x -D uid=ldap-admin,ou=users,\
dc=example,dc=net -W cn=EXAMPLE.NET -LLL
Enter LDAP Password:
dn: cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net
cn: EXAMPLE.NET
objectClass: top
objectClass: krbRealmContainer
objectClass: krbTicketPolicyAux
krbSearchScope: 2
krbSubTrees: ou=verwaltung,dc=example,dc=net
krbSubTrees: ou=produktion,dc=example,dc=net
krbSubTrees: ou=users,dc=example,dc=net

```

Listing 14.65 Suche im LDAP-Baum

Fügen Sie hier alle OUs ein, in denen sich Benutzer befinden. Nur so können Sie sicherstellen, dass sich alle Benutzer über Kerberos authentifizieren können. Wie Sie in dem Beispiel sehen, haben wir hier immer die gesamte Abteilung hinzugefügt. Aber natürlich können


Sie an dieser Stelle auch nur die OU der Benutzer hinzufügen. Sie können auch über den LAM weitere OUs hinzufügen. Das hat den Vorteil, dass nicht jedes Mal die komplette Liste neu geschrieben werden muss. In Abbildung 14.3 sehen Sie die Eigenschaften des Objekts `cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net`.

The screenshot shows the configuration page for the LDAP entry `cn=EXAMPLE.NET`. The DN is `cn=EXAMPLE.NET,cn=kerberos,dc=example,dc=net`. The page includes several attributes and their values:

- cn**: EXAMPLE.NET (required, rdn)
- krbSearchScope**: 2
- krbSubTrees**: ou=verwaltung,dc=example,dc=net; ou=produktion,dc=example,dc=net; ou=users,dc=example,dc=net
- objectClass**: top, krbRealmContainer (structurally), krbTicketPolicyAux

Buttons for "Interne Attribute anzeigen", "Diesen Eintrag löschen", "Diesen DN mit einem anderen vergleichen", and "Neues Attribut hinzufügen" are visible at the top. A "Eintrag aktualisieren" button is at the bottom.

Abbildung 14.3 Anpassen der Suchpfade

Hier sehen Sie, dass es jetzt ein Attribut `krbSubTrees` mit allen eingetragenen OUs gibt. Nach dieser Änderung müssen Sie den *KDC*- und den *kadmin-server* unbedingt neu starten. 

Leider können Sie die alten Passwörter der bestehenden Benutzer nicht in den entsprechenden Principal migrieren. Für alle Benutzer im LDAP-Baum benötigen Sie ein neues Passwort. Der Grund ist, dass die Passwörter verschlüsselt im LDAP liegen. Um sie in das Passwort für den Kerberos umzuwandeln, müsste das Passwort unverschlüsselt vorliegen.

Im Beispiel gibt es den Benutzer `uid=ptau,ou=users,dc=example,dc=net`, der sowohl einen Kerberos-Principal als auch ein LDAP-Objekt besitzt. In Listing 14.66 sehen Sie das Objekt, bevor die Kerberos-Attribute hinzugefügt werden:

```
dn: uid=ptau,ou=users,dc=example,dc=net
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
loginShell: /bin/bash
homeDirectory: /home/ptau
uid: ptau
cn: p tau
userPassword:: e1NTSEF9T3NuZ1dodjl3UXBjeEFoNFhoeGhaS3pZc1ROSlFuSnI=
uidNumber: 10008
gidNumber: 10000
sn: tau
givenName: p
```

Listing 14.66 Reiner LDAP-Benutzer

Um die bestehenden Benutzer, die schon einen Principal im LDAP-Baum besitzen, nun zusammenzuführen, löschen Sie zuerst den alten Principal. Anschließend können Sie das bestehende Benutzerobjekt um die entsprechenden Attribute erweitern. Das realisieren Sie wieder mit dem Kommando `kadmin`, so wie Sie es in Listing 14.67 sehen:

```
kadmin: delete_principal ptau
Sind Sie sicher, dass Sie den Principal »ptau@EXAMPLE.NET« löschen möchten? \
    (yes/no): yes
Principal »ptau@EXAMPLE.NET« gelöscht
Stellen Sie sicher, dass Sie diesen Principal aus allen ACLs entfernt haben, bevor \
    Sie ihn erneut benutzen.

kadmin: add_principal -x dn="uid=ptau,ou=users,dc=example,dc=net" -pw geheim ptau
WARNUNG: Für ptau@EXAMPLE.NET wurde keine Richtlinie angegeben, es wird die \
    Vorgabe »keine Richtlinie« verwendet.
Principal "ptau@EXAMPLE.NET" created.
```

Listing 14.67 Erweitern der bestehenden Benutzer

Nach dem Löschvorgang wurde ein neuer Principal erstellt. Dieser wurde mit dem Parameter `-x dn="uid=ptau,ou=users,dc=example,dc=net"` einem Benutzer im LDAP-Baum zugewiesen. In Listing 14.68 sehen Sie, dass der neue Principal jetzt wieder vorhanden ist:

```
kadmin: listprincs
ptau@EXAMPLE.NET
```

```

stefan@EXAMPLE.NET
K/M@EXAMPLE.NET
krbtgt/EXAMPLE.NET@EXAMPLE.NET
kadmin/admin@EXAMPLE.NET
kadmin/k1-stat.example.net@EXAMPLE.NET
kiprop/k1-stat.example.net@EXAMPLE.NET
kadmin/changepw@EXAMPLE.NET
kadmin/history@EXAMPLE.NET
host/k1-stat.example.net@EXAMPLE.NET
host/k2-stat.example.net@EXAMPLE.NET
ldap/k1-stat.example.net@EXAMPLE.NET
root/admin@EXAMPLE.NET
ldap/provider-stat.example.net@EXAMPLE.NET
host/provider-stat.example.net@EXAMPLE.NET
host/consumer-stat.example.net@EXAMPLE.NET
ldap/consumer-stat.example.net@EXAMPLE.NET

```

Listing 14.68 Neue Liste der Principals

14

In Listing 14.69 werfen wir noch einmal einen Blick auf das Objekt des Benutzers. So können Sie nach und nach jeden bestehenden Benutzer anpassen. Auch hier können Sie wieder den LAM nutzen, in dem Sie zu den Benutzern einfach die Kerberos-Erweiterung hinzufügen.

```

dn: uid=ptau,ou=users,dc=example,dc=net
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: krbPrincipalAux
objectClass: krbTicketPolicyAux
loginShell: /bin/bash
homeDirectory: /home/ptau
uid: ptau
cn: p tau
userPassword:: e0FSR090Mn0kYXJ....
uidNumber: 10001
gidNumber: 10000
sn: tau
givenName: p
structuralObjectClass: inetOrgPerson
entryUUID: 28b74fbc-0c39-103d-94fe-655e5353ea1d
creatorsName: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
createTimestamp: 20221209181509Z
krbLoginFailedCount: 0

```

```
krbPrincipalName: ptau@EXAMPLE.NET
krbPrincipalKey:: MIG2oAMCAQGhAwI....
krbLastPwdChange: 20221209182149Z
krbExtraData:: AAK9fJNjcm9vdc9hZG1pbkBFWEFNUEXFLk5FVAA=
krbExtraData:: AAgBAA==
```

Listing 14.69 Der Benutzer mit Kerberos-Attributen

14.12 Neue Benutzer in den LDAP-Baum aufnehmen

Wenn Sie jetzt neue Benutzer zum LDAP-Baum hinzufügen wollen, realisieren Sie dies immer in zwei Schritten. Im ersten Schritt legen Sie den Benutzer an. Ein Beispiel für eine LDIF-Datei sehen Sie in Listing 14.70:

```
dn: uid=ktom,ou=users,dc=example,dc=net
cn: Kater Tom
gidNumber: 10000
givenName: Kater
homeDirectory: /home/ktom
loginShell: /bin/bash
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
sn: Tom
uid: ktom
uidNumber: 10002
```

Listing 14.70 LDIF-Datei für einen neuen Benutzer

Im zweiten Schritt erweitern Sie den neuen Account um die Kerberos-Attribute. In Listing 14.71 sehen Sie die Vorgehensweise. Wenn Sie den LAM Pro verwenden, können Sie den Benutzer auch in einem Schritt anlegen.

```
kadmin: add_principal -x dn="uid=ktom,ou=users,dc=example,dc=net" -pw geheim ktom
WARNUNG: Für ktom@EXAMPLE.NET wurde keine Richtlinie angegeben, \
        es wird die Vorgabe »keine Richtlinie« verwendet.
Principal "ktom@EXAMPLE.NET" created.
```

```
root@provider-stat:~# ldapsearch -x -D uid=ldap-admin,ou=users,\
        dc=example,dc=net -W uid=ktom -LLL
```

Enter LDAP Password:

```
dn: uid=ktom,ou=users,dc=example,dc=net
cn: Kater Tom
gidNumber: 10000
```

```

givenName: Kater
homeDirectory: /home/ktom
loginShell: /bin/bash
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: krbPrincipalAux
objectClass: krbTicketPolicyAux
sn: Tom
uid: ktom
uidNumber: 10002
krbLoginFailedCount: 0
krbPrincipalName: ktom@EXAMPLE.NET
krbPrincipalKey:: MIG2oAMCAQGHawIBAAIDAgEBo...
krbLastPwdChange: 20221209182545Z
krbExtraData:: AAJGS99fcm9vdc9hZG1pbkBFWEFNUEXFLk5FVAA=
krbExtraData:: AAgBAA==

```

```

root@kerberos01:~# kinit ptau
Passwort für ptau@EXAMPLE.NET:

```

```

root@kerberos01:~# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: ptau@EXAMPLE.NET

```

```

Valid starting      Expires            Service principal
09.12.2022 19:27:59 10.12.2022 05:27:59 krbtgt/EXAMPLE.NET@EXAMPLE.NET
           erneuern bis 10.12.2022 19:27:56

```

Listing 14.71 Anlegen der Kerberos-Attribute

14.13 Authentifizierung am LDAP-Server über »GSSAPI«

Nachdem Sie die Anmeldung der Benutzer auf die Verwendung von Kerberos umgestellt haben, geht es im nächsten Schritt darum, die Authentifizierung am LDAP-Server, für zum Beispiel `ldapsearch`, über das *Generic Security Service Application Programming Interface* (GSSAPI) zu realisieren.

14.13.1 Authentifizierung einrichten

Bevor Sie an dieser Stelle weitermachen, prüfen Sie, ob auf Ihren LDAP-Servern das Paket `krb5-user` installiert ist, denn in diesem Paket befinden sich die benötigten Kommandos.

Sorgen Sie dafür, dass eine */etc/krb5.conf* mit dem Inhalt aus Listing 14.72 vorhanden ist:

```
[libdefaults]
    default_realm = EXAMPLE.NET

[realms]
    EXAMPLE.NET = {
        admin_server = kerberos01.example.net
    }

[domain_realm]
    .example.net = EXAMPLE.NET
```

Listing 14.72 Die »krb5.conf« auf dem LDAP-Server

Diese Datei muss auf allen Kerberos-Clients mit dem selben Inhalt vorhanden sein. Um die Authentifizierung über das GSSAPI durchführen zu können, installieren Sie das Paket *lib-sasl2-modules-gssapi-mit*. Nach der Installation des Pakets starten Sie den LDAP-Server neu. Damit LDAP-Clientprogramme wie *ldapsearch* jetzt auch wissen, dass sie die Authentifizierung ab sofort über Kerberos mit GSSAPI durchführen sollen, fügen Sie noch den Eintrag *SASL_MECH GSSAPI* in der Datei */etc/ldap/ldap.conf* hinzu.

Bei Verwendung der Symas-Pakete

Verwenden Sie für Ihren OpenLDAP-Server die Symas-Pakete, sind die Änderungen etwas umfangreicher. Erstellen Sie als Erstes das Verzeichnis */opt/symas/etc/sasl2*. In diesem Verzeichnis legen Sie eine Datei mit dem Namen *slapd.conf* an. Schreiben Sie die Zeilen aus Listing 14.73 in die Datei:

```
mech_list: gssapi digest-md5 cram-md5 external
keytab: /opt/symas/etc/openldap/ldap.keytab
```

Listing 14.73 Inhalt der Datei »slapd.conf«

Sorgen Sie dafür, dass die Gruppe *openldap* das Leserecht an der Datei */opt/symas/etc/openldap/ldap.keytab* besitzt. Testen Sie anschließend, ob der SASL-Mechanismus *GSSAPI* vorhanden ist. Den Test sehen Sie in Listing 14.74:

```
root@ldap01:~# ldapsearch -x -H ldapi:/// -b "" -LLL -s base supportedSASLMechanisms
dn:
supportedSASLMechanisms: GSSAPI
supportedSASLMechanisms: DIGEST-MD5
supportedSASLMechanisms: CRAM-MD5
supportedSASLMechanisms: EXTERNAL
```

Listing 14.74 Testen des SASL-Mechanismus

Hier werden jetzt alle Mechanismen aus der Datei *slapd.conf* angezeigt. Sollten Sie nicht diese Auflistung erhalten, prüfen Sie die Berechtigungen an der Datei */opt/symas/etc/openldap/ldap.keytab* und starten Sie den LDAP-Dienst neu.

Erste Tests mit GSSAPI

Wenn Sie den LDAP-Server mit allen ACLs eingerichtet haben, dann erhalten Sie beim Versuch, sich das eigene Objekt anzeigen zu lassen, die Meldungen aus Listing 14.75. Wir zeigen hier noch einmal alle Schritte:

```
ktom@ldap01:~$ kinit ktom
Passwort für ktom@EXAMPLE.NET:
root@ldap01:~# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: ktom@EXAMPLE.NET
```

Valid starting	Expires	Service principal
09.12.2022 19:35:45	10.12.2022 05:35:45	krbtgt/EXAMPLE.NET@EXAMPLE.NET
erneuern bis 10.12.2022 19:35:42		

```
ktom@ldap01:~$ ldapsearch -LLL uid=ktom
SASL/GSSAPI authentication started
SASL username: stefan@EXAMPLE.NET
SASL SSF: 256
SASL data security layer installed.
```

```
ktom@provider-stat:~$ ldapwhoami
SASL/GSSAPI authentication started
SASL username: ktom@EXAMPLE.NET
SASL SSF: 256
SASL data security layer installed.
dn:uid=ktom,cn=gssapi,cn=auth
```

```
root@ldap01:~# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: ktom@EXAMPLE.NET
```

Valid starting	Expires	Service principal
09.12.2022 19:35:45	10.12.2022 05:35:45	krbtgt/EXAMPLE.NET@EXAMPLE.NET
erneuern bis 10.12.2022 19:35:42		
09.12.2022 19:42:20	10.12.2022 05:35:45	ldap/ldap01.example.net@EXAMPLE.NET
erneuern bis 10.12.2022 19:35:42		

Listing 14.75 Auflisten des eigenen Objekts

Der Benutzer holt sich ein TGT und sucht dann nach seinem Objekt. Es wird aber nichts angezeigt, obwohl er an seinem Objekt alle Rechte hat. Den Grund sehen Sie in dem Ergebnis von `ldapwhoami`: Der eigentliche DN des Benutzers `uid=ktom,ou=users,dc=example,dc=net` lautet jetzt `uid=ktom,cn=gssapi,cn=auth` und dieser Benutzer hat keine Rechte im LDAP. Die Änderung des DNs geschieht durch die Kerberos-Authentifizierung.

Interessant ist es, wenn Sie nach der Suche einen Blick in die Systemlog-Datei des LDAP-Servers werfen und sich den Ablauf der Authentifizierung und der Suche ansehen. In Listing 14.76 sehen Sie einen Auszug aus dem Log:

```
conn=1009 fd=26 ACCEPT from IP=192.168.56.45:57034 (IP=0.0.0.0:636)
conn=1009 fd=26 TLS established tls_ssf=256 ssf=256 tls_proto=TLSv1.3 \
    tls_cipher=TLS_AES_256_GCM_SHA384
conn=1009 op=0 BIND dn="" method=163
conn=1009 op=0 RESULT tag=97 err=14 qtime=0.000007 etime=0.008403 \
    text=SASL(0): successful result:
conn=1009 op=1 BIND dn="" method=163
conn=1009 op=1 RESULT tag=97 err=14 qtime=0.000007 etime=0.000046 \
    text=SASL(0): successful result:
conn=1009 op=2 BIND dn="" method=163
conn=1009 op=2 BIND authcid="ktom" authzid="ktom"
conn=1009 op=2 BIND dn="uid=ktom,cn=gssapi,cn=auth" mech=GSSAPI \
    bind_ssf=56 ssf=256
conn=1009 op=2 RESULT tag=97 err=0 qtime=0.000006 etime=0.000063 text=
conn=1009 op=3 SRCH base="dc=example,dc=net" scope=2 deref=0 filter="(uid=ktom)"
conn=1009 op=3 SEARCH RESULT tag=101 err=0 qtime=0.000007 etime=0.000101 \
    nentries=0 text=
conn=1009 op=4 UNBIND
conn=1009 fd=26 closed

conn=1010 fd=26 ACCEPT from IP=192.168.56.45:49196 (IP=0.0.0.0:636)
conn=1010 fd=26 TLS established tls_ssf=256 ssf=256 tls_proto=TLSv1.3 \
    tls_cipher=TLS_AES_256_GCM_SHA384
conn=1010 op=0 BIND dn="" method=163
conn=1010 op=0 RESULT tag=97 err=14 qtime=0.000007 etime=0.006597 \
    text=SASL(0): successful result:
conn=1010 op=1 BIND dn="" method=163
conn=1010 op=1 RESULT tag=97 err=14 qtime=0.000006 etime=0.000047 \
    text=SASL(0): successful result:
conn=1010 op=2 BIND dn="" method=163
conn=1010 op=2 BIND authcid="ktom" authzid="ktom"
conn=1010 op=2 BIND dn="uid=ktom,cn=gssapi,cn=auth" mech=GSSAPI \
    bind_ssf=56 ssf=256
conn=1010 op=2 RESULT tag=97 err=0 qtime=0.000006 etime=0.000063 text=
```



```

conn=1010 op=3 EXT oid=1.3.6.1.4.1.4203.1.11.3
conn=1010 op=3 WHOAMI
conn=1010 op=3 RESULT oid= err=0 qtime=0.000007 etime=0.000048 text=
conn=1010 op=4 UNBIND
conn=1010 fd=26 closed

```

Listing 14.76 Auszug aus der Systemlog-Datei

Alle Zeilen, die mit `conn=1009` beginnen, gehören zum Kommando `ldapsearch`, und alle Zeilen die mit `conn=1010` beginnen, zum Kommando `ldapwhoami`.

Hier sehen Sie, dass aus dem Benutzer-DN `uid=ktom,ou=users,dc=example,dc=net` ein Principal `uid=ktom,cn=gssapi,dn=auth` geworden ist. Auch sehen Sie, dass beim `ldapsearch` der Zugriff, auf den LDAP `anonym` geschieht. Das ist aber laut den ACLs nicht erlaubt. Durch die geänderte Authentifizierung können Ihre bisherigen ACLs natürlich nicht mehr greifen. Aber auch dafür gibt es eine Lösung: die `authz-regexp` in der Konfiguration Ihres LDAP-Servers. Über die `olcAuthzRegexp` lassen sich die Principals in LDAP-DNs umschreiben. Da wir in unserem Beispielbaum Benutzer an unterschiedlichen Stellen eingetragen haben, wollen wir Ihnen hier zeigen, wie Sie alle Namen mithilfe einer Regel übersetzen können. Wichtig ist jetzt nur, dass Ihre Principals in Kerberos und die Benutzernamen im LDAP-Baum identisch und eindeutig sind. In Listing 14.77 folgt die LDIF-Datei für die dynamische Konfiguration:

```

dn: cn=config
changetype: modify
add: olcAuthzRegexp
olcAuthzRegexp: {0}uid=(.+),cn=gssapi,cn=auth ldap:///dc=example,dc=net??sub?(uid=$1)

```

Listing 14.77 LDIF-Datei für »authz-regexp«

Auf www.openldap.org/doc/admin26/sasl.html finden Sie eine sehr ausführliche Beschreibung von `olcAuthzRegexp`.

Benutzer mit »cn«

Wenn Sie Benutzer sowohl über das Attribut `uid` als auch über das Attribut `cn` verwalten, legen Sie für das Attribut eine zusätzliche Regel an. Denken Sie auch daran, dass Sie die Übersetzung auf allen LDAP-Servern einrichten – sowohl auf den Providern als auch auf den Consumern.

Noch funktioniert die Übersetzung der SASL-Benutzernamen auf den eigentlichen DN der Benutzer nicht, denn es fehlt noch eine ACL in unserer Konfiguration. Wie schon gesehen, findet die Suche für die Übersetzung immer `anonym` statt. Jetzt ist es aber so, dass wir mit unseren ACLs, die wir im LDAP-Kapitel gesetzt haben, keinen anonymen Zugriff auf den LDAP zulassen. Der Prozess für die Übersetzung benötigt an den Attributen `entry` und `uid`

das Recht auth. Dass der Zugriff anonym stattfindet, sehen Sie auch an den fett gedruckten Zeilen in Listing 14.76. Daher erweitern Sie die ACLs jetzt so wie in Listing 14.78:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
add: olcAccess
olcAccess: {1}to attr=entry,uid
    by anonymous auth
    by * break
```

Listing 14.78 Erweiterte Rechte für die SASL-Übersetzung

Da die Attribute nur für die Authentifizierung genutzt werden können und nie der eigentliche Wert eines Attributs zurückgegeben wird, entsteht dadurch auch keine Sicherheitslücke. Jetzt können Sie die Übersetzung der Namen nochmals mit `ldapwhoami` testen. Dadurch erhalten Sie das Ergebnis aus Listing 14.79:

```
ktom@ldap01:~$ kinit
Passwort für ktom@EXAMPLE.NET:

root@ldap01:~# ldapwhoami
SASL/GSSAPI authentication started
SASL username: ktom@EXAMPLE.NET
SASL SSF: 56
SASL data security layer installed.
dn:uid=ktom,ou=users,dc=example,dc=net
```

Listing 14.79 Erfolgreiche Übersetzung des SASL-Namens

Auch an dieser Stelle ist es eine gute Idee, noch einmal einen Blick in das Log zu werfen. Wie das Ergebnis der Umsetzung aussieht, sehen Sie in Listing 14.80:

```
conn=1034 fd=35 ACCEPT from IP=192.168.56.45:42502 (IP=0.0.0.0:636)
conn=1034 fd=35 TLS established tls_ssf=256 ssf=256 tls_proto=TLSv1.3 \
    tls_cipher=TLS_AES_256_GCM_SHA384
conn=1034 op=0 BIND dn="" method=163
conn=1034 op=0 RESULT tag=97 err=14 qtime=0.000013 etime=0.011261 \
    text=SASL(0): successful result:
conn=1034 op=1 BIND dn="" method=163
conn=1034 op=1 RESULT tag=97 err=14 qtime=0.000010 etime=0.000063 \
    text=SASL(0): successful result:
conn=1034 op=2 BIND dn="" method=163
conn=1034 op=2 BIND authcid="ktom" authzid="ktom"
conn=1034 op=2 BIND dn="uid=ktom,ou=users,dc=example,dc=net" mech=GSSAPI \
    bind_ssf=56 ssf=256
conn=1034 op=2 RESULT tag=97 err=0 qtime=0.000008 etime=0.000272 text=
```

```

conn=1034 op=3 EXT oid=1.3.6.1.4.1.4203.1.11.3
conn=1034 op=3 WHOAMI
conn=1034 op=3 RESULT oid= err=0 qtime=0.000008 etime=0.000059 text=
conn=1034 op=4 UNBIND
conn=1034 fd=35 closed

```

Listing 14.80 »ldapsearch« nach der Umsetzung der Principals

Hier sehen Sie, dass als Bind-DN nach der Suche der echte DN des Benutzers verwendet wird und nicht mehr der SASL-Benutzername. Jetzt greifen auch wieder alle anderen ACLs, und die Benutzer können den LDAP wieder mit ihrem Konto nutzen.

14.13.2 Den zweiten KDC an den LDAP-Server anbinden

Nachdem nun der Kerberos-Server seine Informationen im LDAP-Baum abgelegt hat und dort sucht, sollten Sie im nächsten Schritt einen zweiten Kerberos-Server einbinden, um die Ausfallsicherheit des Dienstes zu gewährleisten. Hier soll jetzt der in Abschnitt 14.9.3 eingerichtete Slave-Kerberos-Server mit an den LDAP-Server angebunden werden. Gehen Sie dazu wie folgt vor:

- ▶ Stoppen Sie den Kerberos-Dienst auf dem Server.
- ▶ Installieren Sie das Paket `krb5-kdc-ldap`.
- ▶ Löschen Sie im nächsten Schritt die lokale Kerberos-Datenbank mit dem Kommando `kdb5_util destroy`.
- ▶ Kopieren Sie die Dateien `/etc/krb5.conf`, `/etc/krb5kdc/service.keyfile` und `/etc/krb5kdc/kdc.conf` vom Master-KDC auf den Slave-KDC.
- ▶ Starten Sie den Kerberos-Dienst neu.

Mit diesen Schritten haben Sie jetzt den Kerberos-Slave-Server ebenfalls an den LDAP gebunden. Der Vorteil dieser Konfiguration ist der, dass Sie so die Replikation der Kerberos-Datenbank nicht mehr über Skripte und den `cron` realisieren müssen. Beide Kerberos-Server greifen auf die LDAP-Datenbank zu, die immer aktuell ist. Die Replikation der Datenbank ist jetzt Aufgabe des LDAP-Servers. Jetzt sind Sie fertig und können den Kerberos-Server nach Ihren Erfordernissen konfigurieren und verwalten. Wie Sie die einzelnen Netzwerkdienste mit dem Kerberos-Server verbinden können, lesen Sie bitte in den Kapiteln zu den jeweiligen Netzwerkdiensten nach.

14.14 Konfiguration des LAM Pro

Wenn Sie Ihre Benutzer nicht nur über die Kommandozeile verwalten wollen, sondern auch über ein grafisches Werkzeug, ist der *LDAP Account Manager (LAM)* eine gute Wahl. Die

LAM-Pro-Version ist zwar nicht kostenfrei, bietet aber die Möglichkeit, Ihre Benutzer komplett grafisch zu verwalten. Um Ihnen einen Einblick in die Möglichkeiten zu geben, wollen wir Ihnen hier die Funktion der Benutzerverwaltung mit dem LAM zeigen.

Nach der Installation des LAM Pro können Sie ihn wie gewohnt konfigurieren. Wenn Sie später auch über den LAM die Kerberos-Passwörter der Benutzer ändern wollen, benötigen Sie auf dem Webserver noch ein paar Anpassungen der Einstellungen. Der Webserver wird später das Programm `kadmin` aufrufen, um die Passwörter zu ändern. Daher benötigt der Webserver auch das Recht dazu. Das Recht erhält der Webserver über einen Service-Principal.

Legen Sie als Erstes einen Principal für den `HTTP`-Service an. Anschließend erzeugen Sie eine `keytab`-Datei, die dann in den Webserver, auf dem der LAM läuft, eingebunden wird. In Listing 14.81 sehen Sie die Vorgehensweise:

```
kadmin: addprinc -randkey host/lam.example.net
WARNUNG: Für host/lam.example.net@EXAMPLE.NET wurde keine Richtlinie angegeben, \
es wird die Vorgabe »keine Richtlinie« verwandt.
Principal "host/lam.example.net@EXAMPLE.NET" created.
kadmin: ktadd -k /root/lam-krb5.keytab host/lam.example.net
Der Eintrag für Principal host/lam.example.net mit KVNO 2 und \
Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
Schlüsseltabelle WRFIL:/root/lam-krb5.keytab hinzugefügt.
Der Eintrag für Principal host/lam.example.net mit KVNO 2 und \
Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
Schlüsseltabelle WRFIL:/root/lam-krb5.keytab hinzugefügt.
kadmin: addprinc -randkey HTTP/lam.example.net
WARNUNG: Für HTTP/lam.example.net@EXAMPLE.NET wurde keine Richtlinie angegeben, \
es wird die Vorgabe »keine Richtlinie« verwandt.
Principal "HTTP/lam.example.net@EXAMPLE.NET" created.
kadmin: ktadd -k /root/lam-service.keytab HTTP/lam.example.net
Der Eintrag für Principal HTTP/lam.example.net mit KVNO 2 und \
Verschlüsselungstyp aes256-cts-hmac-sha384 wurde der \
Schlüsseltabelle WRFIL:/root/lam-service.keytab hinzugefügt.
Der Eintrag für Principal HTTP/lam.example.net mit KVNO 2 und \
Verschlüsselungstyp aes128-cts-hmac-sha256-128 wurde der \
Schlüsseltabelle WRFIL:/root/lam-service.keytab hinzugefügt.
```

Listing 14.81 Erstellen des Service-Keys

Kopieren Sie die beiden `keytab`-Dateien auf den Webserver. Wenn Sie den Webserver für den LAM auf einem der LDAP-Server einsetzen, benötigen Sie keine neue Host-`keytab`-Datei; die besitzt der LDAP-Server bereits.

Geben Sie dem Webserver noch das Recht, Passwörter zu ändern. Das regeln Sie über die Datei `kadm5.acl` auf dem Kerberos-Admin-Server, wie in Listing 14.82:

```
*/admin@EXAMPLE.NET *
HTTP/lam.example.net@EXAMPLE.NET c
*@EXAMPLE.NET il
*/*@EXAMPLE.NET i
```

Listing 14.82 Anpassung der ACLs

Installation von kadmin auf dem Webserver

Wenn Sie den Webserver getrennt vom LDAP- und Kerberos-Server betreiben, installieren Sie jetzt das Paket *krb5-user* auf dem Webserver. Denn der Webserver benötigt das Programm *kadmin* für die Änderung der Passwörter, und das Kommando ist Bestandteil des Pakets. Kopieren Sie ebenfalls eine *krb5.conf* auf den Webserver.

Starten Sie anschließend den KDC und den Admin-Server neu, um die Änderungen wirksam werden zu lassen. Jetzt können Sie das Ändern der Passwörter auf dem Webserver testen, so wie Sie es in Listing 14.83 sehen:

```
root@lam:~# kadmin -k -t /etc/apache2/lam-service.keytab -p HTTP/lam.example.net \
-q 'cpw -pw supergeheim ptau'
Authentifizierung als Principal HTTP/lam.example.net mit Schlüsseltabelle \
/etc/apache2/lam-service.keytab
Passwort von »ptau@EXAMPLE.NET« geändert
```

Listing 14.83 Änderung der Passwörter

Erst wenn Sie diesen Schritt erfolgreich ausführen können, dürfen Sie mit der Konfiguration des LAM fortfahren. Sorgen Sie jetzt dafür, dass die Gruppe, unter der der Webserver läuft, das Leserecht an der *keytab*-Datei erhält. Passen Sie auch die Berechtigungen an der Log-Datei für den *kadmin* so an, dass die Gruppe, unter der der Webserver läuft, dort schreiben darf. Beim Erstellen der Datei hat nur der *root* Rechte an der Datei.

Jetzt können Sie das Profil für Ihren LDAP-Server im LAM anpassen. Dafür ändern Sie in der Profilverwaltung unter dem Karteireiter **MODULEINSTELLUNGEN** den **BEFEHL FÜR PASSWORTÄNDERUNG**. In Abbildung 14.4 sehen Sie eine Beispieleinstellung. Die Einstellungen für unser Beispiel lauten `/usr/sbin/kadmin -k -t /etc/apache2/lam-service.keytab -p HTTP/lam.example.net@EXAMPLE.NET`



Abbildung 14.4 Einstellung der Passwortänderung im »LAM Pro«

Damit beim Anlegen eines neuen Benutzers der Realm richtig gesetzt wird, melden Sie sich als Admin Ihres LDAP-Servers am LAM an, starten den PROFILEDITOR und passen die Einstellungen für das Kerberos-Modul so an, wie Sie es in Abbildung 14.5 sehen.

The screenshot shows a user profile editor interface with the following sections:

- Home Directory and Shell:**
 - Heimatverzeichnis: /home/\$user
 - Login Shell: /bin/bash
- Shadow (Password Policy):**
 - Diese Erweiterung automatisch hinzufügen:
 - Passwortwarnung: 10
 - Passwortablauf: 10
 - Minimales Passwortalter: 1
 - Maximales Passwortalter: 365
 - Ablaufdatum: Three dropdown menus, all set to '-'
- Kerberos:**
 - Bereich: EXAMPLE.NET
 - Ticketlaufzeit: (empty field)
 - Laufzeitverlängerung: (empty field)

At the bottom, there are two buttons: "Speichern" (Save) and "Abbrechen" (Cancel).

Abbildung 14.5 Einstellung in der Profilverwaltung des »LAM Pro«

Wenn Sie jetzt einen neuen Benutzer anlegen, können Sie das Kerberos-Modul für den Benutzer aktivieren und anschließend das Kerberos-Passwort ändern. Das UNIX-Passwort müssen Sie nicht setzen. Nachdem Sie den Benutzer angelegt haben, können Sie ihn mit `kinit` testen.

Damit haben Sie neben der Verwaltung der Benutzer über die Kommandozeile auch die Möglichkeit, Ihre Benutzer grafisch zu verwalten.

Jetzt haben Sie eine Umgebung, die aus zwei LDAP-Servern und zwei Kerberos-Servern besteht, die ihre Daten alle im LDAP ablegen. Sie können die Benutzer sowohl auf der Kommandozeile verwalten als auch über ein grafisches Werkzeug.

Kapitel 15

Samba 4

In diesem Kapitel geht es um die Samba 4-Administration. Als Beispiel soll eine Active Directory-Domäne mit zwei Domaincontrollern eingerichtet werden. Zusätzlich wird ein Fileserver in die Domäne integriert. Auch die Verwaltung von Clients in der Domäne wird in diesem Kapitel angesprochen.

Mit Samba 4 können Sie eine Active Directory-Struktur in Ihrem Netzwerk implementieren, ohne einen Windows-Server einsetzen zu müssen. Samba 4 wird dann die Rolle des Domaincontrollers übernehmen. Sie können auch mehrere Samba 4-Domaincontroller betreiben, um die Ausfallsicherheit der Anmeldesysteme zu erhöhen. Mithilfe der Standortverwaltung können Sie die Anmeldung in verschiedene Bereiche unterteilen, um eventuelle WAN-Verbindungen zwischen den Standorten nicht durch die Anmeldungen zu überlasten.

Die Möglichkeit, mit Gruppenrichtlinien den Zugriff auf Ressourcen oder Einstellungen zu beschränken, haben Sie auch mit einem Samba-Domaincontroller. Sie können hier alle Gruppenrichtlinien verwenden, die Sie eventuell schon aus der Windows-Welt kennen.

15.1 Vorüberlegungen

Zunächst überlegen Sie, welche Dienste Sie einsetzen wollen. Dann: Soll eine Active-Directory-Domäne aufgesetzt oder nur ein Standalone Server oder ein Memberserver in der Domäne eingerichtet werden? Denn davon hängt es ab, welche Distribution Sie einsetzen können: Noch können nicht alle Distributionen die Funktion eines Active Directory-Domaincontrollers bereitstellen. Der Grund ist der verwendete Heimdal-Kerberos von Samba 4.

Seit der Samba-Version 4.7 wird zwar der *MIT-Kerberos* unterstützt, noch ist aber der Heimdal-Kerberos Standard. Und erst wenn der MIT-Kerberos komplett implementiert und getestet ist, werden auch die anderen Distributionen den Domaincontroller unterstützen. Selbst in der aktuellen Samba-Version 4.17 ist der MIT-Kerberos immer noch *Experimental*. Wir hier im Buch nutzen die Samba-Version 4.17.3, diese Version erhalten Sie unter anderem in den Backports von Debian 11. Da Red Hat und somit auch Suse keine Installation als Domaincontroller zulassen – Grund dafür ist der fehlende MIT-Kerberos-Support von Samba, werden wir in diesem Kapitel ausschließlich Debian und Ubuntu nutzen. Die Pakete von Louis van Belle haben wir hier leider entfernen müssen, da zum Zeitpunkt, als wir dieses Buches überarbeitet haben, keine aktuellen Pakete mehr von ihm bereitgestellt werden.

Welche Windows-Version nutzen wir? Zu dem Zeitpunkt, als wir das Kapitel überarbeitet haben, war Windows 10 noch vorherrschend und Windows 11 noch kaum im Einsatz, aus dem Grund haben wir viele Bilder in diesem Kapitel noch auf Windows 10 erstellt, nur dort, wo es gravierende Abweichungen gibt, haben wir Bilder für beide Versionen erstellt. Bei der Verwaltung der Domäne, ist die Darstellung unter beiden Versionen identisch.

Die unterschiedlichen Samba-Versionen stellen unterschiedliche Dienste und Möglichkeiten bereit. Achten Sie darauf, dass nur die jeweils letzten drei Versionen direkt vom Samba-Team unterstützt und weiterentwickelt werden. Beim Schreiben dieses Buches waren das die Versionen 4.17, 4.16 und 4.15. Wenn Ihre Distribution nur eine ältere Version unterstützt, sind Sie darauf angewiesen, dass die Hersteller der Distribution sicherheitskritische Patches einarbeiten.

Eine andere Möglichkeit, um immer aktuelle Pakete für Samba in einer Produktivumgebung nutzen zu können und eventuell Support zu erhalten, sind die Pakete der Firma Sernet. Die Pakete sind nicht kostenfrei, aber sehr stabil und für sehr viele Distributionen erhältlich. Mithilfe der Sernet-Pakete können Sie auch openSUSE und CentOS als Domaincontroller einsetzen, da Sernet auch diese Pakete inklusive des passenden Kerberos-Servers bereitstellt.

Tragen Sie die *bullseye*-Backports in Ihre Datei `/etc/apt/sources.list` ein, und führen Sie anschließend ein `apt update` aus. Danach können Sie die benötigten Pakete installieren. Wollen Sie den Kerberos-Server später testen, installieren Sie zusätzlich das Paket *heimdal-clients*. Alle Abhängigkeiten werden automatisch aufgelöst. In Listing 15.1 sehen Sie den Vorgang der Installation für ein Debian-System:

```
root@addc-01:~# apt install -t bullseye-backports samba libpam-heimdal heimdal-\
clients ldb-tools winbind libpam-winbind smbclient libnss-winbind bind9 dnsutils
```

Listing 15.1 Installation der Pakete

Während der Installation der Pakete werden Sie nach dem Kerberos-Realm gefragt. An dieser Stelle können Sie die Abfrage einfach mit bestätigen. Die Informationen werden für die Datei `/etc/krb5.conf` benötigt, die beim Provisioning des Domaincontrollers passend für Ihre Domäne erstellt wird. Die Datei kopieren Sie dann nur noch an die richtige Stelle.

15.2 Konfiguration von Samba 4 als Domaincontroller

Nach der Installation der Pakete geht es jetzt darum, den ersten Domaincontroller zu konfigurieren. Für die Konfiguration benötigen Sie die folgenden Informationen, die während der Konfiguration abgefragt werden:

► den Realm

Der Realm wird für den Kerberos-Server benötigt. Der Realm wird bei der Einrichtung des DNS-Servers auch als DNS-Domainname verwendet.



Achtung: Verwenden Sie nie die TLD .local

Immer wieder sehen wir den Fall, dass als Domäne die TLD `.local` verwendet wird. Dies ist ein schwerer Fehler, denn `.local` wird von vielen Betriebssystemen als Multicast-Domäne verwendet, um IP-Adressen aus dem Bereich 169.254.0.1 bis 169.254.255.254 zu vergeben. Das geschieht immer dann, wenn ein Client keinen DHCP-Server kontaktieren kann. Bei einer Namensauflösung des Clients wird dann bei einer Anfrage, die sich an eine `.local`-Domäne richtet, eine Multicast-IP aus dem Bereich 224.x.x.x generiert. Mehr zu dem Thema finden Sie unter https://de.wikipedia.org/wiki/Zeroconf#Multicast_DNS.

► **den NetBIOS-Domainnamen**

Der NetBIOS-Domainname ist die Adresse, über die der Server per NetBIOS-Protokoll erreichbar ist. Der NetBIOS-Name sollte immer der erste Teil des Realms sein.

► **die Funktion des Servers**

Sie sollten wissen, welche Rolle der Server in der Domäne übernehmen soll. In unserem Fall übernimmt er die Rolle des Domaincontrollers.

► **den DNS-Server, den Sie verwenden möchten**

Entscheiden Sie sich, ob Sie den internen DNS-Server von Samba4 oder den *Bind9*-Server nutzen wollen. Für unser Beispiel werden wir den *Bind9* als Nameserver verwenden. Nur mit dem *Bind9* können Sie einen *Round-Robin*-DNS-Server einrichten. Diese Option benötigen Sie immer dann, wenn Sie zusätzlich noch einen Cluster als Fileserver installieren wollen. Denn dann wird über das DNS-Round-Robin das Loadbalancing durchgeführt.

► **die IP-Adresse eines eventuell benötigten DNS-Forwarders**

An diese IP-Adresse werden alle DNS-Anfragen weitergeleitet, die nicht zur eigenen Zone gehören. Ohne einen Forwarder ist keine Namensauflösung von Internetnamen möglich. Nur bei der Verwendung des internen Nameservers von Samba wird diese Information benötigt – beim *Bind9* erfolgt die Konfiguration der Weiterleitung in *named.conf.options*.

Damit Sie die Übersicht behalten, haben wir Ihnen die Werte, die wir im weiteren Verlauf dieses Kapitels verwenden werden, in Tabelle 15.1 aufgelistet.

Parameter	Wert
Realm	EXAMPLE.NET
NetBIOS-Domainname	EXAMPLE
Funktion	Domaincontroller
DNS-Server	BIND9_DLZ

Tabelle 15.1 Übersicht über die SMB-Protokollversionen



Bevor Sie mit dem Provisioning beginnen, achten Sie darauf, dass die Datei `/etc/hosts` die richtigen Einträge enthält. In Listing 15.2 sehen Sie die relevanten Zeilen:

```
127.0.0.1      localhost
192.168.56.201 addc-01.example.net addc-01
```

Listing 15.2 Einträge in der Datei »`/etc/hosts`«

Tragen Sie keine weiteren IP-Adressen in die Datei ein. Durch falsche Einträge in der Datei `/etc/hosts` kann es zu Fehlern bei der Namensauflösung kommen. Für die Konfiguration und die Administration eines Samba 4-Servers steht Ihnen das Kommando `samba-tool` zur Verfügung. Mit diesem Kommando können Sie die Domäne einrichten und verwalten, aber auch später die Benutzer und Gruppen sowie die Gruppenrichtlinien und den DNS-Server verwalten.

In Listing 15.3 sehen Sie eine Übersicht über die Aufgaben in Ihrer Domäne, die Sie mit dem Kommando `samba-tool` durchführen können:

Main samba administration tool.

Options:

```
-h, --help      show this help message and exit
```

Version Options:

```
-V, --version  Display version number
```

Available subcommands:

```
computer      - Computer management.
contact       - Contact management.
dbcheck       - Check local AD database for errors.
delegation    - Delegation management.
dns           - Domain Name Service (DNS) management.
domain        - Domain management.
drs           - Directory Replication Services (DRS) management.
dsacl         - DS ACLs manipulation.
forest        - Forest management.
fsmo          - Flexible Single Master Operations (FSMO) roles management.
gpo           - Group Policy Object (GPO) management.
group         - Group management.
ldapcmp       - Compare two ldap databases.
ntacl         - NT ACLs manipulation.
ou            - Organizational Units (OU) management.
processes     - List processes (to aid debugging on systems without setproctitle).
rodc          - Read-Only Domain Controller (RODC) management.
schema        - Schema querying and management.
```

```

sites      - Sites management.
spn        - Service Principal Name (SPN) management.
testparm   - Syntax check the configuration file.
time       - Retrieve the time on a server.
user       - User management.
visualize  - Produces graphical representations of Samba network state.

```

Listing 15.3 Möglichkeiten mit dem Kommando »samba-tool«

Immer wenn Sie das Kommando `samba-tool` mit einem der Subkommandos angeben, ohne weitere Parameter zu verwenden, wird eine Hilfe zu dem entsprechenden Subkommando angezeigt. Zu jedem Subkommando können Sie dann noch über den Parameter `-h` zusätzliche Hilfen erhalten.

15.2.1 Das Provisioning

Löschen der `smb.conf`

Bei der Installation der Samba-Pakete wird bereits eine `/etc/samba/smb.conf` installiert. Diese ist aber auf einem Domaincontroller absolut unbrauchbar und führt sogar zu Fehlern beim Provisioning. Aus diesem Grund löschen Sie die Datei `/etc/samba/smb.conf` vor dem Provisioning.



15

Starten Sie jetzt das *Provisioning* auf dem ersten Domaincontroller. Dieser Vorgang ist für alle Distributionen identisch. In Listing 15.4 sehen Sie den Ablauf der Konfiguration:

```

root@addc-01:~# samba-tool domain provision
Realm [EXAMPLE.NET]:
Domain [EXAMPLE]:
Server Role (dc, member, standalone) [dc]:
DNS backend (SAMBA_INTERNAL, BIND9_FLATFILE, BIND9_DLZ, NONE) [SAMBA_INTERNAL]: \
  BIND9_DLZ
Administrator password:
Retype password:
[...]
See /var/lib/samba/bind-dns/named.conf for an example configuration
  include file for BIND
and /var/lib/samba/bind-dns/named.txt for further documentation required \
  for secure DNS updates
Setting up sam.ldb rootDSE marking as synchronized
Fixing provision GUIDs
A Kerberos configuration suitable for Samba AD has been generated at \
  /var/lib/samba/private/krb5.conf

```

```
Merge the contents of this file with your system krb5.conf or replace \
  it with this one. Do not create a symlink!
Once the above files are installed, your Samba AD server will be ready to use
Server Role:      active directory domain controller
Hostname:        addc-01
NetBIOS Domain:  EXAMPLE
DNS Domain:      example.net
DOMAIN SID:      S-1-5-21-2492729547-2181408212-3299912776
```

Listing 15.4 Das Provisioning des ersten Domaincontrollers



Aus Gründen der Übersichtlichkeit haben wir hier einige Meldungen nicht mit aufgelistet, sondern uns auf das Wesentliche beschränkt. Achten Sie aber besonders auf die Hinweise zur Verwendung des *Bind9* und zur Datei */var/lib/samba/private/krb5.conf*.

Kopieren Sie die Datei */var/lib/samba/privat/krb5.conf* noch direkt in das Verzeichnis */etc*. Bei dieser Datei handelt es sich um die Kerberos-Client-Datei. Über die Informationen in dieser Datei findet der Server die entsprechenden Service-Records. Wie Sie in Listing 15.4 sehen können, wird jetzt der Bind9 als DNS-Server verwendet. Sorgen Sie dafür, dass der Bind9 auch auf die Datenbanken des Active Directories zugreifen kann und dass Sie einen *Forwarder* für die Namensauflösung außerhalb Ihrer Domäne eingetragen haben. Um den Bind9 zu konfigurieren, wurden beim *Provisioning* bereits einige Dateien erzeugt, sodass die Einrichtung des Bind9 recht einfach zu realisieren ist.

15.2.2 Konfiguration des Bind9

Um den Bind9 zu konfigurieren, suchen Sie die beiden Dateien *named.conf.options* und *named.conf.local* im Verzeichnis */etc/bind*. In beiden Dateien werden Änderungen für den Einsatz im Active Directory benötigt. In Listing 15.5 sehen Sie die Anpassungen für die Datei *named.conf.options*:

```
forwarders {
    8.8.8.8;
};
tkey-gssapi-keytab "/var/lib/samba/bind-dns/dns.keytab";
```

Listing 15.5 Die neue »named.conf.options«-Datei

Entfernen Sie die Kommentarzeichen vor der Konfiguration der Forwarder, und tragen Sie die IP-Adresse Ihres DNS-Forwarders ein. Ermöglichen Sie noch die Kerberos-Authentifizierung für den Bind9. Dafür verweisen Sie auf die *dns.keytab*-Datei, die während des Provisionings erstellt wurde. Ohne diese Datei kann der Bind9 sich nicht am Active Directory authentifizieren und damit nicht auf die DNS-Zonen zugreifen. Im Anschluss passen Sie die Datei *named.conf.local* an. Ergänzen Sie die Datei so wie in Listing 15.6 dargestellt.

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
include "/var/lib/samba/bind-dns/named.conf";
```

Listing 15.6 Einträge in der Datei »named.conf.local«

Wie Sie sehen, wird hier nur eine zusätzliche Zeile eingetragen. Diese verweist auf eine Konfiguration, die im Samba-Verzeichnis liegt und während des Provisionings passend zur installierten Bind9-Version angepasst wird. Zonendateien werden bei der Verwendung von Bind9 im Active Directory mit Samba 4 nicht einrichtet, denn die Datenbanken mit den entsprechenden Records wurden während des Provisionings bereits erstellt. Alle DNS-Daten befinden sich im Active Directory.

Bei früheren Versionen war es notwendig, die Berechtigungen im Verzeichnis `/var/lib/samba/bind-dns` zu prüfen und eventuell anzupassen: Zu den Problemen kam es aufgrund eines Bugs, der aber mittlerweile behoben ist. Heute stimmen die Rechte.



15

Sowohl unter Ubuntu als auch unter Debian wird der Bind9 über *AppArmor* in eine Art *chroot*-Umgebung gezwungen. Das heißt, der Bind9 kann nur auf die Dateien zugreifen, die von AppArmor freigegeben wurden. Damit der Bind überhaupt Zugriff auf die Datenbanken des Active Directory bekommen kann, passen Sie die Konfiguration von AppArmor in der Datei `/etc/AppArmor.d/usr.sbin.named` so wie in Listing 15.7 an:

```
/etc/bind/** r,
/var/lib/bind/** rw,
/var/lib/bind/ rw,
/var/cache/bind/** lrw,
/var/lib/samba/** rwmk,
/usr/lib/x86_64-linux-gnu/** rwmk,
/dev/urandom rwmk,
```

Listing 15.7 Einstellungen von AppArmor

Bei den aktuellen Paketen wird diese Anpassung automatisch vorgenommen Sie brauchen hier in den meisten Fällen keine Änderungen vorzunehmen.



Jetzt können Sie den Bind9 neu starten und anschließend in der Log-Datei nach den Zeilen schauen, die den erfolgreichen Zugriff auf das Active Directory melden:

```
root@addc-01:~# systemctl restart bind9
[...]
```

```
root@addc-01:~# tail -n 200 /var/log/syslog
[...]
...: samba_dlz: Loading 'AD DNS Zone' using driver dlopen
...: samba_dlz: started for DN DC=example,DC=net
...: samba_dlz: starting configure
...: samba_dlz: configured writeable zone 'example.net'
...: samba_dlz: configured writeable zone '_msdcs.example.net'
```

Listing 15.8 Erster Start des Bind9

Erst wenn Sie im Log-File diese Einträge finden, ist der Bind9 vollständig konfiguriert und hat alle Rechte, um auf das Active Directory zugreifen zu können. Samba verwendet unterschiedliche Dienste abhängig davon, welche Funktion der Samba-Server übernehmen soll. Die Standardeinstellung ist dabei *Memberserver*. Da Sie jetzt aber einen Domaincontroller starten wollen, sorgen Sie dafür, dass die Dienste für den Memberserver deaktiviert und die Dienste für den Domaincontroller aktiviert sind (siehe Listing 15.9):

```
root@addc-01:~# systemctl stop smbd nmbd winbind
root@addc-01:~# systemctl disable smbd nmbd winbind
Synchronizing state of smbd.service with SysV service script with \
  /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable smbd
Synchronizing state of nmbd.service with SysV service script with \
  /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable nmbd
Synchronizing state of winbind.service with SysV service script with \
  /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable winbind
Removed /etc/systemd/system/multi-user.target.wants/winbind.service.
Removed /etc/systemd/system/multi-user.target.wants/smbd.service.
Removed /etc/systemd/system/multi-user.target.wants/nmbd.service.

root@addc-01:~# systemctl unmask samba-ad-dc
Removed /etc/systemd/system/samba-ad-dc.service.

root@addc-01:~# systemctl enable samba-ad-dc
Synchronizing state of samba-ad-dc.service with SysV service script with \
  /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable samba-ad-dc
Created symlink /etc/systemd/system/multi-user.target.wants/samba-ad-dc.\
  service -> /lib/systemd/system/samba-ad-dc.service.

root@addc-01:~# systemctl start samba-ad-dc
```

Listing 15.9 Aktivieren der Domaincontroller-Funktion



Verwendug der Sernet-Pakete

Wenn Sie die Sernet-Pakete nutzen, wird die Entscheidung, welche Rolle der Samba-Server übernehmen soll, über die Datei `/etc/default/sernet-samba` getroffen. Tragen Sie dort `ad` bei der Startart ein, damit die Funktion des Domaincontrollers unterstützt wird. Anschließend starten Sie den Dienst mit dem Kommando `systemctl start sernet-samba-ad-dc`.

Anschließend können Sie mit dem Kommando `ps ax | grep samba` prüfen, ob der Samba-Domaincontroller-Dienst gestartet wurde. In Listing 15.10 sehen Sie das Ergebnis:

```
root@addc-01:~# ps ax | grep samba
5117 ?      Ss   0:00 samba: root process
5118 ?      S    0:00 samba: tfork waiter process(5119)
5119 ?      S    0:00 samba: task[s3fs] pre-fork master
5120 ?      S    0:00 samba: tfork waiter process(5121)
5121 ?      S    0:00 samba: task[rpc] pre-fork master
5122 ?      S    0:00 samba: tfork waiter process(5124)
5123 ?      S    0:00 samba: tfork waiter process(5126)
5124 ?      S    0:00 samba: task[nbt] pre-fork master
5125 ?      S    0:00 samba: tfork waiter process(5127)
5127 ?      S    0:00 samba: task[wrepl] pre-fork master
5128 ?      S    0:00 samba: tfork waiter process(5130)
5129 ?      S    0:00 samba: tfork waiter process(5132)
5130 ?      S    0:00 samba: task[ldap] pre-fork master
5131 ?      S    0:00 samba: tfork waiter process(5134)
5132 ?      S    0:00 samba: task[rpc] pre-forked worker(0)
5133 ?      S    0:00 samba: tfork waiter process(5136)
5134 ?      S    0:00 samba: task[cldap] pre-fork master
5135 ?      S    0:00 samba: tfork waiter process(5138)
5136 ?      S    0:00 samba: task[rpc] pre-forked worker(1)
5137 ?      S    0:00 samba: tfork waiter process(5140)
5138 ?      S    0:00 samba: task[kdc] pre-fork master
5139 ?      S    0:00 samba: tfork waiter process(5142)
5140 ?      S    0:00 samba: task[rpc] pre-forked worker(2)
5141 ?      S    0:00 samba: tfork waiter process(5147)
5142 ?      S    0:00 samba: task[drepl] pre-fork master
5143 ?      S    0:00 samba: tfork waiter process(5145)
5144 ?      S    0:00 samba: tfork waiter process(5148)
5145 ?      S    0:00 samba: task[winbindd] pre-fork master
5146 ?      S    0:00 samba: tfork waiter process(5149)
5147 ?      S    0:00 samba: task[kdc] pre-forked worker(0)
5148 ?      S    0:00 samba: task[rpc] pre-forked worker(3)
5149 ?      S    0:00 samba: task[ntp_signd] pre-fork master
```

```
5150 ?      S      0:00 samba: tfork waiter process(5157)
5151 ?      S      0:00 samba: tfork waiter process(5152)
5152 ?      S      0:00 samba: task[kdc] pre-forked worker(1)
5153 ?      S      0:00 samba: tfork waiter process(5155)
5154 ?      S      0:00 samba: tfork waiter process(5156)
5155 ?      S      0:00 samba: task[kdc] pre-forked worker(2)
5157 ?      S      0:00 samba: task[kcc] pre-fork master
5158 ?      S      0:00 samba: tfork waiter process(5160)
5159 ?      S      0:00 samba: tfork waiter process(5161)
5160 ?      S      0:00 samba: task[dnupdate] pre-fork master
5161 ?      S      0:00 samba: task[kdc] pre-forked worker(3)
5175 ?      S      0:00 samba: tfork waiter process(5176)
5176 ?      S      0:00 samba: task[ldap] pre-forked worker(0)
5177 ?      S      0:00 samba: tfork waiter process(5178)
5178 ?      S      0:00 samba: task[ldap] pre-forked worker(1)
5179 ?      S      0:00 samba: tfork waiter process(5180)
5180 ?      S      0:00 samba: task[ldap] pre-forked worker(2)
5181 ?      S      0:00 samba: tfork waiter process(5182)
5182 ?      S      0:00 samba: task[ldap] pre-forked worker(3)
5186 pts/0   S+    0:00 grep samba
```

Listing 15.10 Erster Start des Domaincontrollers

Passen Sie jetzt noch die Konfiguration der Netzwerkumgebung so an, dass der Domaincontroller nur sich selbst zur Namensauflösung nutzen kann. Wenn Sie auf Ihrem System das Paket *resolvconf* installiert haben, passen Sie am einfachsten die Datei */etc/network/interfaces* so wie in Listing 15.11 an:

```
allow-hotplug enp0s3
iface enp0s3 inet static
    address 192.168.56.201
    netmask 255.255.255.0
    gateway 192.168.56.254
    dns-nameservers 192.168.56.81
    dns-search example.net
```

Listing 15.11 Anpassung des Resolvers

Bevor Sie die Funktionstests durchführen, starten Sie die Server einmal komplett neu. So können Sie sicher sein, dass alle Dienste auch bei einem Systemstart sauber starten.

15.3 Testen des Domaincontrollers

Nach dem Neustart können Sie mit zwei einfachen Kommandos testen, ob der Samba-Domaincontroller auch richtig gestartet wurde. Den ersten Test haben Sie bereits vor dem

Neustart des Systems durchgeführt, indem Sie die Prozesse mit dem Kommando `ps ax | grep samba` aufgelistet haben. Führen Sie das Kommando noch einmal aus, um zu sehen, ob die Liste der Prozesse identisch ist. Anschließend prüfen Sie mit dem Kommando `ss` die offenen Ports. Die Kommandos und die Ausgabe des Kommandos `ss` sehen Sie in Listing 15.12:

```
root@addc-01:~# ps ax | grep samba
[...]
root@addc-01:~# ss -tln | awk 'print $1" "$2" "$3" "$4'
State Recv-Q Send-Q Local
LISTEN 0 10 192.168.56.201:53
LISTEN 0 10 10.0.2.15:53
LISTEN 0 10 127.0.0.1:53
LISTEN 0 128 0.0.0.0:22
LISTEN 0 10 0.0.0.0:88
LISTEN 0 128 127.0.0.1:953
LISTEN 0 10 0.0.0.0:636
LISTEN 0 50 0.0.0.0:445
LISTEN 0 10 0.0.0.0:49152
LISTEN 0 10 0.0.0.0:49153
LISTEN 0 10 0.0.0.0:49154
LISTEN 0 10 0.0.0.0:3268
LISTEN 0 10 0.0.0.0:3269
LISTEN 0 10 0.0.0.0:389
LISTEN 0 10 0.0.0.0:135
LISTEN 0 50 0.0.0.0:139
LISTEN 0 10 0.0.0.0:464
LISTEN 0 10 [::]:53
LISTEN 0 128 [::]:22
LISTEN 0 10 [::]:88
LISTEN 0 128 [::1]:953
LISTEN 0 10 [::]:636
LISTEN 0 50 [::]:445
LISTEN 0 10 [::]:49152
LISTEN 0 10 [::]:49153
LISTEN 0 10 [::]:49154
LISTEN 0 10 [::]:3268
LISTEN 0 10 [::]:3269
LISTEN 0 10 [::]:389
LISTEN 0 10 [::]:135
LISTEN 0 50 [::]:139
LISTEN 0 10 [::]:464
```

Listing 15.12 Tests nach dem Neustart

Im Test mit dem Kommando `ss` sehen Sie die entsprechenden Ports für alle Dienste, die der Samba-Domaincontroller bereitstellt. Hier fällt auf, dass der Samba sowohl auf IPv4- als auch auf IPv6-Anfragen reagiert. Wollen Sie dieses Verhalten unterbinden, können Sie in der Datei `/etc/samba/smb.conf` die Einträge für die `interfaces` wie in Listing 15.13 vornehmen:

```
[global]
    netbios name = ADDC-01
    realm = EXAMPLE.NET
    server role = active directory domain controller
    server services = s3fs, rpc, nbt, wrepl, ldap, cldap, kdc, drepl, \
                    winbindd, ntp_signd, kcc, dnsupdate
    workgroup = EXAMPLE
    interfaces = 192.168.56.81
    bind interfaces only = yes

[netlogon]
    path = /var/lib/samba/sysvol/example.net/scripts
    read only = No

[sysvol]
    path = /var/lib/samba/sysvol
    read only = No
```

Listing 15.13 Anpassung an der »smb.conf«

Starten Sie anschließend den Dienst neu. Jetzt reagiert der Samba-DC nur noch auf Anfragen über die eine IPv4-Adresse. Das können Sie wieder mit dem Kommando `ss` prüfen.

15.3.1 Testen des DNS-Servers

Im nächsten Test überprüfen Sie, ob Ihr Domaincontroller die Einstellung für den Name-server richtig übernommen hat und ob der Nameserver die Namen richtig auflöst. In Listing 15.14 sehen Sie verschiedene Tests:

```
root@addc-01:~# host addc-01
addc-01.example.net has address 192.168.56.81

root@addc-01:~# host -t SRV _kerberos._tcp.example.net
_kerberos._tcp.example.net has SRV record 0 100 88 addc-01.example.net.

root@addc-01:~# host -t SRV _ldap._tcp.example.net
_ldap._tcp.example.net has SRV record 0 100 389 addc-01.example.net.

root@addc-01:~# host -t SRV _gc._tcp.example.net
_gc._tcp.example.net has SRV record 0 100 3268 addc-01.example.net.
```

```
root@addc-01:~# host 192.168.56.81
Host 201.56.168.192.in-addr.arpa. not found: 3(NXDOMAIN)
```

Listing 15.14 Test des DNS-Servers

Ein Reverse-Lookup funktioniert noch nicht, da bei der Einrichtung der Domäne lediglich die Forward-Zone eingerichtet wird. Die Reverse-Zone legen Sie immer von Hand an. Um den Domaincontroller auch rückwärts auflösen zu können, sehen Sie in Listing 15.15 die Kommandos zur Erstellung der Zone und zum Anlegen des Records für den ersten Domaincontroller:

```
root@addc-01:~# kinit administrator
administrator@EXAMPLE.NET's Password:
root@addc-01:~# samba-tool dns zonecreate addc-01 56.168.192.in-addr.arpa -N
Zone 56.168.192.in-addr.arpa created successfully
root@addc-01:~# samba-tool dns add addc-01 56.168.192.in-addr.arpa \
81 PTR addc-01.example.net
Record added successfully
root@addc-01:~# host 192.168.56.81
81.56.168.192.in-addr.arpa domain name pointer addc-01.example.net.
```

Listing 15.15 Anlegen der Reverse-Zone

Der Parameter `-k yes` wird nicht mehr benötigt, da das Kerberos-Ticket automatisch erkannt wird. Sie können das Kommando auch ohne `-N` eingeben, dann werden Sie, trotz Ticket, nach einem Passwort gefragt. An der Stelle können Sie dann aber eingeben, was immer Sie wollen: Es funktioniert immer, denn es wird immer das Ticket überprüft.

Bei einer Revers-Zone unbedingt beachten!

Zu jeder DNS-Zone gehört immer mindestens ein NS-Record. Der NS-Record zeigt immer mindestens auf einen Nameserver, der diese Zone bereitstellt. Im Gegensatz zu einer neuen *Forward-Zone*, bei der jeder neue Domaincontroller als NS-Record eingetragen wird, wird in eine *Reverse-Zone* nur der Domaincontroller eingetragen, auf dem Sie die Zone eingerichtet haben. Problematisch wird das, wenn Sie diesen Domaincontroller aus der Domäne entfernen, denn dann wird auch der NS-Record der Zone gelöscht und Sie haben dann eine Zone ohne *NS-Record*. Das führt dazu, dass der *Bind9* nicht mehr startet. Sorgen Sie deshalb dafür, dass immer mindestens zwei Domaincontroller als NS-Record für alle Ihre Reverse-Zonen eingetragen sind.



15.3.2 Test des Verbindungsaufbaus

Jetzt können Sie den Verbindungsaufbau zum Samba-4-Server testen. In Listing 15.16 sehen Sie den Test des Verbindungsaufbaus mit dem Kommando `smbclient`:

```
root@addc-01:~# smbclient -L addc-01
Password for [EXAMPLE\root]:<RETURN>
Anonymous login successful
```

Sharename	Type	Comment
-----	----	-----
sysvol	Disk	
netlogon	Disk	
IPC\$	IPC	IPC Service (Samba 4.17.3-Debian)

```
SMB1 disabled -- no workgroup available
```

Listing 15.16 Test des Verbindungsaufbaus

In Listing 15.16 sehen Sie, dass bereits zwei Freigaben auf dem Domaincontroller bereitgestellt werden: `sysvol` und `netlogon`. Diese beiden Freigaben werden auf einem Domaincontroller immer benötigt und somit bei der Erstkonfiguration auch immer angelegt. Die Verwendung der beiden Freigaben werden wir im weiteren Verlauf dieses Kapitels noch genauer vorstellen und erläutern. Außerdem sehen Sie, dass das Protokoll SMB1 nicht aktiv ist. Somit ist eine Einbindung von Windows-XP-Clients in eine neue Domäne nicht mehr möglich. Aus Sicherheitsgründen sollten Sie das Protokoll auch möglichst nicht mehr aktivieren. Seit der Version 4.17 ist es möglich, den Samba ganz ohne die Unterstützung von SMBv1 zu kompilieren. In Zukunft wird die Protokollversion ganz entfernt werden.

Kommen wir an dieser Stelle noch einmal zum Kerberos-Server zurück. Sie haben sich ja bereits ein Ticket mit `kinit administrator` für den Domain-Admin gezogen und damit die neuen Einträge im DNS angelegt. Im Beispiel aus Listing 15.16 wurde noch kein Ticket vom Kerberos bezogen, aus diesem Grund werden Sie hier auch nach dem Passwort für den gerade angemeldeten Benutzer (den `root`) gefragt. Wenn Sie hier keine Passwort angeben, findet die Anfrage anonym statt. Haben Sie aber bereits ein Kerberos-Ticket für einen Benutzer vom KDC erhalten, werden Sie nach dem Passwort des Benutzers gefragt, dem das TGT-Ticket gehört. Die aktuellen Versionen des `smb`-Kommandos erkennen, wenn ein Kerberos-Ticket vorhanden ist, und nutzen dann auch sofort Kerberos für die Authentifizierung. Die Passwortabfrage können Sie über den Parameter `-N` unterdrücken. Dann sehen Sie sofort das Ergebnis. Bei der Passwortabfrage trotz vorhandenem Ticket, können Sie eingeben was Sie wollen, es klappt immer, denn es wird ausschließlich das Ticket genutzt. In Listing 15.17 sehen Sie den Unterschied zu der anonymen Anmeldung:

```
root@addc-01:~# smbclient -L addc-01 -N
```

Sharename	Type	Comment
-----	----	-----
sysvol	Disk	
netlogon	Disk	

```
IPC$          IPC          IPC Service (Samba 4.17.3-Debian)
SMB1 disabled -- no workgroup available
```

Listing 15.17 Abfrage mit Kerberos-Ticket

Beim nächsten Zugriff auf den Dienst brauchen Sie dem Dienst nicht mehr mitzuteilen, dass eine Authentifizierung via Kerberos durchgeführt werden soll. Vielmehr erkennt das Programm, dass Sie bereits ein Serviceticket besitzen, und verwendet es für die Authentifizierung. Alle Tickets können Sie sich mit dem Kommando `klist` anzeigen lassen. Listing 15.18 zeigt Ihnen die Tickets, die bis zu dem Zeitpunkt für den Administrator gespeichert wurden:

```
root@addc-01:~# klist
Credentials cache: FILE:/tmp/krb5cc_0
Principal: administrator@EXAMPLE.NET

Issued                Expires              Principal
Dec 19 14:13:31 2022  Dec 20 00:13:31 2022  krbtgt/EXAMPLE.NET@EXAMPLE.NET
Dec 19 14:13:38 2022  Dec 20 00:13:31 2022  cifs/addc-01@EXAMPLE.NET
```

Listing 15.18 Alle Kerberos-Tickets des Administrators

Authentifizieren Sie sich wenn immer möglich über Kerberos und nicht mit Ihrem Benutzernamen und Passwort. Im Verlauf des Buches werden wir Ihnen nur noch die Kommandos für die Verwaltung des Samba-Servers mit Kerberos-Authentifizierung zeigen.

Wenn Sie Kerberos verwenden, wird ein Host oder ein Benutzer nicht erneut bei jedem Zugriff authentifiziert. Vielmehr wird der Benutzer sein Ticket, das er von dem Kerberos-Server erhalten hat, dazu nutzen, um sich bei dem Dienst zu authentifizieren.

Löschen aller Tickets

Denken Sie daran: Immer wenn Sie auf Ihrer Konsole ein Ticket für den Administrator gezogen haben, kann jeder, der Zugriff auf Ihr Terminal hat, dieses Ticket nutzen. Wenn Sie sich ein Ticket als Domain-Admin gezogen haben und Ihren Arbeitsplatz verlassen wollen, sollten Sie entweder das Terminal sperren oder entfernen Sie alle Tickets mit dem Kommando `kdestroy`. Damit werden alle Tickets gelöscht und für eine Anmeldung bei den Samba-Diensten wird wieder ein Passwort benötigt.



15.3.3 Einrichtung des Zeitserver

Jeder Windows-Client wird bei einer Anmeldung immer die Zeit vom Domaincontroller abfragen, denn die Zeit ist in einer Domäne ein wichtiger Faktor. Dienste wie Kerberos benötigen eine einheitliche Zeit, um die Authentifizierung durchführen zu können. Zu große Zeitabweichungen zwischen einem Client und einem Domaincontroller sorgen dafür, dass

die Authentifizierung fehlschlägt. Aus diesem Grund ist es sehr wichtig, dass Sie auf allen Domaincontrollern immer einen Zeitserver einrichten. Installieren Sie dafür das Paket *ntp*, und ersetzen Sie die Datei */etc/ntp.conf* durch die Zeilen aus Listing 15.19:

```
server 127.127.1.0
fudge 127.127.1.0 stratum 10
server 0.pool.ntp.org iburst prefer
server 1.pool.ntp.org iburst prefer
driftfile /var/lib/ntp/ntp.drift
logfile /var/log/ntp
ntpsigndsocket /var/lib/samba/ntp_signd/
restrict default kod nomodify notrap nopeer mssntp
restrict 127.0.0.1
restrict 0.pool.ntp.org mask 255.255.255.255 nomodify notrap nopeer noquery
restrict 1.pool.ntp.org mask 255.255.255.255 nomodify notrap nopeer noquery
```

Listing 15.19 Konfiguration des Zeitservers

Die beiden Zeitserver von *ntp.org* können Sie auch durch einen eigenen Zeitserver in Ihrem lokalen Netz ersetzen. Damit ein Client die Zeit, die Ihr Zeitserver auf Anfrage sendet, auch akzeptiert, müssen die Informationen vom Domaincontroller signiert werden. Für diese Aufgabe gibt es einen extra bereitgestellten Socket im Verzeichnis */var/lib/samba/ntp_signd*. Mit diesem verbindet sich der Zeitserver, um die Informationen zu signieren.

Damit der Zeitserver auf den Socket in diesem Verzeichnis zugreifen kann, setzen Sie die Berechtigungen so wie in Listing 15.20:

```
root@adminbuch:~# chgrp ntp /var/lib/samba/ntp_signd/
root@adminbuch:~# chmod 750 /var/lib/samba/ntp_signd/
```

Listing 15.20 Die Rechte für den Zeitserver setzen

Erst jetzt können Sie den Zeitserver neu starten. Damit ist der erste Domaincontroller eingerichtet und Sie können mit der Benutzerverwaltung beginnen.

15.4 Benutzer- und Gruppenverwaltung

Für die Benutzerverwaltung unter Samba 4 gibt es verschiedene Wege, die wir Ihnen nun näher vorstellen möchten:

► **mit dem Samba-4-Werkzeug »samba-tool«**

Mit dem *samba-tool* können Sie Benutzer und Gruppen über die Kommandozeile verwalten. Damit haben Sie dann auch die Möglichkeit, mehrere Benutzer über Skripte anzulegen, zu ändern oder zu löschen.

► **über einen Windows-Client mit den RSAT**

Für Windows können Sie die *Windows Remote Server Administration Tools* (RSAT) verwenden. Bei den aktuellen Windows 10- und Windows 11-Versionen benötigen Sie keinen Download der RSAT mehr, denn dort werden die RSAT als Funktion bereitgestellt, die Sie lediglich aktivieren müssen.

► **mit dem LDAP Account Manager (LAM)**

Dank des Einsatzes von Roland Gruber, dem Entwickler des LAM, gibt es ein Modul für den LAM, mit dem Sie die Benutzer und Gruppen von Samba 4 über das webbasierte Werkzeug verwalten können, obwohl bei Samba 4 kein OpenLDAP zum Einsatz kommt, sondern ein eigener LDAP-Server.

Mehr zum LAM

Die Verwendung des LAM zur Verwaltung der Benutzer und Gruppen kann hier nicht weiter erklärt werden, da dies den Rahmen dieses Buches sprengen würde. Eine sehr gute Anleitung zur Einrichtung des *LDAP Account Managers* finden Sie aber unter folgender URL: <https://www.ldap-account-manager.org/lamcms/documentation>



15.5 Benutzer- und Gruppenverwaltung über die Kommandozeile

In Abschnitt 15.5.1 und Abschnitt 15.5.2 geht es um die Verwaltung der Benutzer und Gruppen über die Kommandozeile. Die gesamte Verwaltung der Benutzer und Gruppen erfolgt hier über das Kommando `samba-tool`. Das `samba-tool` ist die Zusammenfassung der unter Samba 3 bekannten `net-Tools` und des Kommandos `pdbedit` und ersetzt diese bei der Verwaltung von Gruppen und Benutzern vollständig. Als Linux-Administrator werden Sie anfangs versuchen, alle Benutzer und Gruppen über die Kommandozeile zu verwalten, was theoretisch möglich ist. Aber Sie werden sehr schnell feststellen, dass es an manchen Stellen einfacher ist, Benutzer über die RSAT zu verwalten (siehe Abschnitt 15.6) – besonders, wenn es darum geht, die Vielzahl an Attributen eines Benutzers zu verändern. Als Windows-Administrator werden Sie hingegen schnell die Vorzüge der Verwaltung der Benutzer und Gruppen über die Kommandozeile schätzen lernen. Das gilt besonders dann, wenn Sie mehrere Benutzer auf einmal anlegen wollen, denn dann können Sie das Kommando `samba-tool` recht einfach in Shell-Skripten einsetzen. Deshalb folgt zunächst eine ausführliche Erklärung der Gruppen- und Benutzerverwaltung über die Kommandozeile mit aussagekräftigen Beispielen. In Abschnitt 15.5.3 befassen wir uns dann mit Passwortrichtlinien.

15.5.1 Verwaltung von Gruppen über die Kommandozeile

Mit dem Kommando `samba-tool group` verwalten Sie die Gruppen. Zu dem Kommando gibt es die verschiedenen Optionen für die Verwaltung. Wenn Sie auf der Kommandozeile nur

das Kommando `samba-tool group` eingeben, dann erhalten Sie eine Hilfe zu dem Kommando. In den folgenden Abschnitten werden alle Subkommandos erläutert, aber nicht in der Reihenfolge, wie sie in der Hilfe aufgelistet sind. So haben Sie die Möglichkeit, alle Beispiele direkt auszuprobieren.

Auflisten der Gruppen mit »group list«

Eine Übersicht über alle Gruppen im System erhalten Sie, wie in Listing 15.21 zu sehen, mit `samba-tool group list`:

```
root@addc-01:~# samba-tool group list
Administrators
Schema Admins
Enterprise Read-only Domain Controllers
Domain Users
Domain Controllers
Guests
Domain Admins
Event Log Readers
Distributed COM Users
DnsAdmins
DnsUpdateProxy
Pre-Windows 2000 Compatible Access
Denied RODC Password Replication Group
Replicator
Domain Guests
Remote Desktop Users
Certificate Service DCOM Access
Allowed RODC Password Replication Group
RAS and IAS Servers
Windows Authorization Access Group
Performance Log Users
Cert Publishers
Performance Monitor Users
Group Policy Creator Owners
Terminal Server License Servers
Enterprise Admins
Cryptographic Operators
Incoming Forest Trust Builders
Network Configuration Operators
Backup Operators
Users
Domain Computers
Print Operators
```



```
IIS_IUSRS
Server Operators
Account Operators
Read-only Domain Controllers
```

Listing 15.21 Auflisten der Gruppen

Hier sehen Sie eine Liste von Gruppen, die nach der Installation des Systems vorhanden sind. Bei diesen Gruppen handelt es sich um Gruppen, die auch für die Verwaltung des AD unter Windows benötigt werden.

Löschen Sie keine der Gruppen

Löschen Sie keine der hier aufgelisteten Gruppen aus Ihrem System! Alle diese Gruppen haben eine feste Bedeutung in der Windows-Welt und werden immer mit einem festen *Security Identifier* (SID) verwaltet. Wenn Sie eine der Gruppen löschen, kann das dazu führen, dass Sie Ihre Domäne neu aufsetzen müssen.

**Auflisten der Gruppenmitglieder einer Gruppe mit »group listmembers <group>«**

Wenn Sie wissen wollen, welche Benutzer Mitglied einer Gruppe sind, können Sie, wie in Listing 15.22 zu sehen, dies mit `samba-tool group listmembers <group>` überprüfen:

```
root@addc-01:~# samba-tool group listmembers administrators
Administrator
Enterprise Admins
Domain Admins
```

Listing 15.22 Auflisten der Gruppenmitglieder

Beim Auflisten der Gruppe `administrators` sehen Sie, dass die Gruppe `Domain Admins` Mitglied der Gruppe ist. Samba4 kann mit verschachtelten Gruppen umgehen.

Auch Sie können später bei der Administration Gruppen verschachteln. Anders als bei Samba3 müssen Sie die Möglichkeit der verschachtelten Gruppen nicht mehr in der Datei `smb.conf` aktivieren.

Anlegen einer neuen Gruppe mit »group add <groupname>«

Eine neue Gruppe können Sie mit dem Kommando `samba-tool group add <groupname>` zu Ihrer Gruppenliste hinzufügen. Listing 15.23 zeigt das Anlegen einer neuen Gruppe:

```
root@addc-01:~# samba-tool group add datengruppe
Added group datengruppe
```

Listing 15.23 Anlegen einer neuen Gruppe

Die gerade angelegte Gruppe ist eine reine Windows-Gruppe. Sie können die Gruppe mit dem Kommando `wbinfo -g` sehen, aber im Moment noch nicht mit `getent group`. In Listing 15.24 sehen Sie die Liste der Gruppen:

```
root@adminbuch:~# wbinfo -g
EXAMPLE\cert publishers
EXAMPLE\ras and ias servers
EXAMPLE\allowed rodc password replication group
EXAMPLE\denied rodc password replication group
EXAMPLE\dnsadmins
EXAMPLE\enterprise read-only domain controllers
EXAMPLE\domain admins
EXAMPLE\domain users
EXAMPLE\domain guests
EXAMPLE\domain computers
EXAMPLE\domain controllers
EXAMPLE\Schema Admins
EXAMPLE\enterprise admins
EXAMPLE\group policy creator owners
EXAMPLE\read-only domain controllers
EXAMPLE\dnsupdateproxy
EXAMPLE\datengruppe
```

Listing 15.24 Liste der Gruppen

Auch können Sie mit `chgrp <neue Gruppe> <Eintrag>` keine Berechtigungen setzen, denn für die Verwendung der Gruppe unter Linux ist es notwendig, dass Sie das ID-Mapping aktivieren. Hier wird zwischen dem ID-Mapping auf einem Domaincontroller und dem ID-Mapping auf einem Fileserver oder einem Linux-Client unterschieden. Auf dem Domaincontroller übernimmt der *winbind* des Samba 4 selbst das ID-Mapping und weist den Windows-Benutzern und -Gruppen eigene IDs zu. Die IDs von Benutzern und Gruppen beginnen immer mit dem Wert 3.000.000 und können auf den Domaincontrollern unterschiedlich sein.

Auf Fileservern oder Linux-Clients übernimmt der *winbind* diese Aufgabe. Mehr zu dieser Problematik erfahren Sie in Abschnitt 15.8, »Linux-Clients in der Domäne«, und in Abschnitt 15.9, »Zusätzliche Server in der Domäne«. Um die Gruppen auch im Linux-System sehen und nutzen zu können, passen Sie die Datei */etc/nsswitch.conf* so wie in Listing 15.25 an:

```
passwd  compat systemd winbind
group   compat systemd winbind
```

Listing 15.25 Anpassen der Datei »nsswitch.conf«

Nach der Anpassung der Datei */etc/nsswitch.conf* können Sie jetzt mit dem Kommando `getent group` nicht alle Gruppen sehen, denn die automatische Auflistung aller Benutzer aus dem Active Directory ist in der Konfiguration immer abgeschaltet.

systemd nicht entfernen

Bei Ubuntu und Debian wird zusätzlich zu dem Parameter `compat` der Parameter `systemd` verwendet. Diesen Parameter dürfen Sie nicht löschen, da sonst bestimmte Systemdienste nicht mehr richtig funktionieren. Fügen Sie diesen Parameter aber bei keiner anderen Distribution hinzu.



Das sollten Sie auch nicht ändern, da eine Auflösung der IDs aller Benutzer und Gruppen einen Domaincontroller sehr stark belastet. Sie können aber mit `getent passwd <ad-user>` oder `getent group <ad-group>` immer die Informationen der Benutzer und Gruppen aus dem Active Directory abfragen. Die Vergabe der Rechte an die Benutzer und Gruppen über die Kommandozeile funktioniert auch.

Wenn Sie später auf dem Domaincontroller keine Daten speichern oder keine Verwaltung der Rechte über die Kommandozeile vornehmen wollen, brauchen Sie die Datei `/etc/nsswitch.conf` nicht anzupassen. Nutzen Sie einen Domaincontroller nie zum Speichern von Daten. Richten Sie für die Speicherung von Daten immer einen Fileserver ein. Nicht nur aus Sicherheitsgründen: Auch das ID-Mapping ist auf jedem Domaincontroller immer unterschiedlich, und somit können Linux-Benutzer eventuell unterschiedliche UIDs auf dem Domaincontroller haben.



15

Hinzufügen eines oder mehrerer Benutzer zu einer bestehenden Gruppe

Über das Kommando `samba-tool group addmembers <groupname> <members>` können Sie mehrere Benutzer gleichzeitig zu einer Gruppe hinzufügen. Listing 15.26 zeigt dieses Vorgehen:

```
root@addc-01:~# samba-tool group addmembers datengruppe "Domain Users"
Added members to group datengruppe
root@addc-01:~# samba-tool group listmembers datengruppe
Domain Users
```

Listing 15.26 Gruppenmitglieder hinzufügen

Da Sie Gruppen verschachteln können, können Sie auch eine oder mehrere der Standardgruppen zu Ihrer Gruppe hinzufügen. Achten Sie darauf, dass einige der Gruppen ein Leerzeichen im Namen haben. Dann müssen Sie den Gruppennamen beim Hinzufügen in Hochkommata setzen.

Keine Verwendung von lokalen Gruppennamen

Verwenden Sie für neue Gruppen keine Namen, die in der lokalen Gruppenverwaltung über die Datei `/etc/group` Verwendung finden. Die lokalen Gruppen haben immer Priorität vor den



Gruppen aus dem AD. Wenn Sie jetzt also eine Gruppe im AD anlegen, die denselben Namen hat wie eine lokale Gruppe, wird das System bei der Rechtevergabe immer die lokale Gruppe verwenden.



Sie können mit dem Kommando `samba-tool group addmembers <groupname> <members>` keine lokalen Gruppen des Systems zu den AD-Gruppen hinzufügen, da diese Gruppen nur auf dem System vorhanden sind und nicht im AD.

Entfernen eines oder mehrerer Benutzer aus einer Gruppe

Wenn Sie einen oder mehrere Benutzer aus einer Gruppe entfernen möchten, geht das mit dem Kommando `samba-tool group removemembers <groupname> <members>`. In Listing 15.27 sehen Sie ein Beispiel:

```
root@addc-01:~# samba-tool group removemembers datengruppe "Domain Users"
Removed members from group datengruppe
```

Listing 15.27 Entfernen von Mitgliedern

Sie können hier auch mehrere Mitglieder, durch Leerzeichen getrennt, aus der Gruppe entfernen.

15.5.2 Verwaltung von Benutzern über die Kommandozeile

Für die Verwaltung der Benutzer verwenden Sie das Kommando `samba-tool user`. Genau wie bei der Verwaltung der Gruppen gibt es auch hier wieder Subkommandos für die verschiedenen Aufgaben. Auch hier werden alle Subkommandos näher erläutert, sodass Sie die Beispiele gleich testen können.

Auflisten der Benutzer

Alle Benutzer können Sie sich mit dem Kommando `samba-tool user list` anzeigen lassen. In Listing 15.28 sehen Sie eine Liste aller Benutzer nach der Installation des Systems:

```
root@addc-01:~# samba-tool user list
Administrator
krbtgt
Guest
dns-addc-01
```

Listing 15.28 Auflistung aller Benutzer

Wie schon zuvor bei den Gruppen sehen Sie hier alle Benutzer, die während der Installation angelegt werden. Auch hier gilt: Löschen Sie keinen der Benutzer!

Anlegen eines Benutzers

Um einen neuen Benutzer über die Kommandozeile anzulegen, verwenden Sie das Kommando `samba-tool user create username <password>`. Achten Sie bei dem Passwort auf die Komplexitätsregel. In Listing 15.29 sehen Sie ein Beispiel mit einem Passwort, das diesen Regeln nicht entspricht:

```
root@adminbuch:~# samba-tool user create Stefan geheim --given-name=Stefan \
--surname=Kania
ERROR(lldb): Failed to add user 'Stefan': - 0000052D: \
Constraint violation - check_password_restrictions: the \
password is too short. It should be equal or longer than 7 characters!
```

Listing 15.29 Passwort, das nicht den Komplexitätsregeln entspricht

Komplexitätsregeln bei Passwörtern

Es müssen mindestens Groß- und Kleinbuchstaben sowie Zahlen verwendet werden oder aber zumindest ein Sonderzeichen. Verwenden Sie immer drei verschiedene Zeichengruppen beim Passwort. Die Mindestlänge eines Passworts ist sieben Zeichen.



15

Alle Benutzer, die Sie über die Kommandozeile anlegen, werden immer in der Organisationseinheit `cn=Users,DC=example,DC=net` erzeugt, wenn Sie keine andere OU angeben. Wenn Sie später eine komplexe AD-Struktur haben, verschieben Sie die neuen Benutzer auf jeden Fall immer in eine von Ihnen erstellte OU, denn auf `cn=users` können Sie keine Gruppenrichtlinie vergeben. In Listing 15.30 sehen Sie das erfolgreiche Anlegen eines neuen Benutzers:

```
root@adminbuch:~# samba-tool user create Stefan Pa55w0rd \
--given-name=Stefan --surname=Kania
User 'Stefan' created successfully
root@addc-01:~# samba-tool user list
Administrator
krbtgt
Stefan
Guest
dns-addc-01
```

Listing 15.30 Erfolgreiches Anlegen eines Benutzers

Wie Sie in dem Beispiel sehen, können Sie beim Anlegen des Benutzers gleich weitere Parameter mit angeben. In diesem Beispiel sind es der Vor- und der Nachname. Alle Werte, die im AD verwendet werden, können hier mit übergeben werden. Da es sich dabei um eine größere Anzahl von Parametern handelt, kann an dieser Stelle nicht auf alle eingegangen werden. Über das Kommando `samba-tool user create --help` können Sie sich anzeigen las-

sen, welche Attribute Sie beim Anlegen eines neuen Benutzers vergeben können. Bei der Benutzerverwaltung mit grafischen Werkzeugen werden Sie alle Parameter sehen und anpassen können. In Listing 15.31 sehen Sie, wie Sie einen Benutzer ohne weitere Parameter anlegen können:

```
root@addc-01:~# samba-tool user create ktom
New Password:
Retype Password:
User 'ktom' created successfully
```

Listing 15.31 Ein weiterer Benutzer

Da dieses Mal kein Passwort beim Anlegen des Benutzers mitgegeben wurde, wird jetzt nach dem Passwort für den Benutzer gefragt. Das Heimatverzeichnis des Benutzers wird nicht mit angelegt. Legen Sie das Heimatverzeichnis des Benutzers auf dem entsprechenden Server von Hand an, und vergeben Sie die benötigten Rechte. Später können Sie das Anlegen der Heimatverzeichnisse über eine Gruppenrichtlinie durchführen. Nach dem Anlegen des Benutzers können Sie sich den Benutzer wieder mit `samba-tool user list` auflisten lassen. Auch die Benutzer sehen Sie wieder mit `wbinfo -u`. Wie schon bei den Gruppen werden die neuen Benutzer mit `getent passwd` nur angezeigt, wenn Sie die Datei `/etc/nsswitch.conf` angepasst haben. In Listing 15.32 sehen Sie sowohl das Ergebnis von `wbinfo -u` als auch das Ergebnis von `getent passwd <username>`:

```
root@addc-01:~# wbinfo -u
EXAMPLE\administrator
EXAMPLE\guest
EXAMPLE\krbtgt
EXAMPLE\dns-addc-01
EXAMPLE\stefan
EXAMPLE\ktom
root@addc-01:~# getent passwd EXAMPLE\stefan
EXAMPLE\stefan:*:3000019:100::/home/EXAMPLE/stefan:/bin/false
```

Listing 15.32 Auflisten der Benutzer

Deaktivieren eines Benutzers

Wenn Sie einen bestehenden Benutzer nur deaktivieren wollen, weil der Benutzer sich zurzeit nicht anmelden darf oder soll, können Sie den Benutzer einfach deaktivieren und müssen ihn nicht gleich löschen. Wenn Sie einen Benutzer löschen, löschen Sie damit auch seinen *SID*, der nicht mehr wiederhergestellt werden kann. Selbst wenn Sie einen neuen Benutzer mit demselben Namen anlegen, ist er für das System ein komplett neuer Benutzer. Aus diesem Grund kann es sinnvoller sein, einen Benutzer erst einmal zu deaktivieren; löschen können Sie ihn immer noch. In Listing 15.33 sehen Sie ein Beispiel für das Deaktivieren eines Benutzers:

```
root@addc-01:~# samba-tool user disable stefan
```

Listing 15.33 Deaktivieren eines Benutzers

Leider erhalten Sie bei `samba-tool user disable <username>` keine Meldung. Ob das so gewollt ist oder ob das ein Bug ist, wird sich noch herausstellen.

Aktivieren eines deaktivierten Benutzers

Um einen zuvor deaktivierten Benutzer wieder zu aktivieren, verwenden Sie das Kommando `samba-tool user enable <username>` so, wie Sie es in Listing 15.34 sehen können. Anders als beim Befehl `disable` bekommen Sie eine Meldung, dass der Benutzer wieder aktiviert wurde.

```
root@addc-01:~# samba-tool user enable Stefan
Enabled user 'Stefan'
```

Listing 15.34 Aktivieren eines Benutzers

Das Passwort eines Benutzers ändern

Für den Fall, dass ein Benutzer sein Passwort vergessen hat, oder wenn Sie einem Benutzer aus einem anderen Grund ein neues Passwort zuweisen wollen, verwenden Sie das Kommando `samba-tool user setpassword <username>`. In Listing 15.35 sehen Sie ein Beispiel dafür. Beim Setzen eines neuen Passworts müssen Sie wieder auf die Komplexitätsregeln achten.

```
root@addc-01:~# samba-tool user setpassword Stefan
New Password:
Retype Password:
Changed password OK
```

Listing 15.35 Das Passworts eines Benutzers setzen

Ändern des persönlichen Passworts durch den Benutzer

Natürlich kann ein Benutzer sein Passwort auch selbst über die Kommandozeile verändern. Dafür gibt es das Kommando `samba-tool user password` (siehe Listing 15.36):

```
EXAMPLE\stefan@addc-01:~# samba-tool user password
Password for [EXAMPLE\stefan]:
New Password:
Retype Password:
Changed password OK
```

Listing 15.36 Änderung des Passworts durch den Benutzer

Löschen eines Benutzers

Wenn Sie einen Benutzer aus dem System entfernen wollen, nutzen Sie dafür das Kommando `user delete <username>` wie in Listing 15.37:

```
root@addc-01:~# samba-tool user delete stefan
Deleted user stefan
```

Listing 15.37 Löschen eines Benutzers

Denken Sie daran, dass ein eventuell vorhandenes Heimatverzeichnis des Benutzers nicht automatisch gelöscht wird. Beim Löschen eines Benutzers wird auch der *SID* gelöscht, und zwar endgültig. Selbst wenn Sie den Benutzer mit den identischen Werten neu anlegen, wird er für das System immer ein komplett neuer Benutzer sein. Aus diesem Grund ist es manchmal sinnvoll, einen Benutzer erst einmal zu deaktivieren – löschen können Sie den Benutzer später immer noch.

15.5.3 Setzen der Passworrichtlinien

Mit dem Kommando `samba-tool` können Sie auch die *Passworrichtlinien* für die Domäne setzen. Bevor Sie die Regeln ändern, lassen Sie sich die aktuellen Richtlinien so auflisten, wie Sie es in Listing 15.38 sehen:

```
root@addc-01:~# samba-tool domain passwordsettings show
Password information for domain 'DC=example,DC=net'
Password complexity: on
Store plaintext passwords: off
Password history length: 24
Minimum password length: 7
Minimum password age (days): 1
Maximum password age (days): 42
Account lockout duration (mins): 30
Account lockout threshold (attempts): 0
Reset account lockout after (mins): 30
```

Listing 15.38 Auflisten der Passworrichtlinien

Wenn Sie einzelne Einstellungen ändern möchten, können Sie sich mit dem Kommando `samba-tool domain passwordsettings set --help` die Hilfe anzeigen lassen, um die Parameter für die einzelnen Optionen bestimmen zu können. Anschließend können Sie die Werte so wie in Listing 15.39 anpassen. Alle Einstellungen sind, wie auch in einer Microsoft-Domäne, für alle Benutzer der Domäne bindend.

```
root@addc-01:~# samba-tool domain passwordsettings set --min-pwd-age=2
Minimum password age changed!
All changes applied successfully!
root@addc-01:~# samba-tool domain passwordsettings set --max-pwd-age=90
Maximum password age changed!
All changes applied successfully!
root@addc-01:~# samba-tool domain passwordsettings show
```



```

Password information for domain 'DC=example,DC=net'
Password complexity: on
Store plaintext passwords: off
Password history length: 24
Minimum password length: 7
Minimum password age (days): 2
Maximum password age (days): 90
Account lockout duration (mins): 30
Account lockout threshold (attempts): 0
Reset account lockout after (mins): 30

```

Listing 15.39 Ändern der Passwortparameter

15.5.4 Passwortrichtlinien mit Password Settings Objects (PSO)

Wollen Sie für unterschiedliche Benutzer unterschiedliche Passwortregeln festlegen, können Sie das über die *Password Settings Objects (PSO)* realisieren. Über die PSO können Sie einzelnen Benutzern oder Gruppen neue Richtlinien zuweisen. Wenn Sie diese Möglichkeit nutzen wollen, empfiehlt es sich, die PSO immer für Gruppen zu erstellen und nicht für einzelne Benutzer. Stattdessen erstellen Sie eine Gruppe, weisen der Gruppe die PSO zu und machen die entsprechenden Benutzer zu Mitgliedern der Gruppe. So können Sie, wenn Sie mehrere Gruppen mit PSO anlegen, Benutzern schnell andere Richtlinien zuweisen, indem Sie sie einfach in eine andere Gruppe verschieben.

Anlegen eines PSO

Beim Anlegen eines PSO vergeben Sie einen Namen für das PSO und eine Priorität (*precedence*) für das PSO. Wenn ein Benutzer in mehreren PSO-Gruppen Mitglied ist, werden immer die Passwortrichtlinien des PSO mit der geringeren Priorität verwendet. Wenn Sie ein PSO erstellen, wird immer mindestens ein Wert für das PSO erwartet. In Listing 15.40 sehen Sie die Erstellung eines neuen PSO:

```

root@addc-01:~# samba-tool domain passwordsettings pso \
    create verwaltung 1 --min-pwd-age=5
Not all password policy options have been specified.
For unspecified options, the current domain password settings \
    will be used as the default values.
PSO successfully created: CN=verwaltung,CN=Password Settings \
    Container,CN=System,DC=example,DC=net
Password information for PSO 'verwaltung'

Precedence (lowest is best): 1
Password complexity: on
Store plaintext passwords: off

```

```
Password history length: 24
Minimum password length: 7
Minimum password age (days): 5
Maximum password age (days): 90
Account lockout duration (mins): 30
Account lockout threshold (attempts): 0
Reset account lockout after (mins): 30
```

Listing 15.40 Erstellen eines PSO

Zuweisen eines PSO

Wie schon eingangs erwähnt wurde, ist es sinnvoll, ein PSO immer einer Gruppe und nicht einem Benutzer zuzuweisen. Wie Sie ein PSO einer Gruppe zuweisen, sehen Sie in Listing 15.41:

```
root@addc-01:~# samba-tool domain passwordsettings pso apply verwaltung datengruppe
PSO 'verwaltung' applied to 'datengruppe'
```

Listing 15.41 Zuweisung eines PSO

Jeder Benutzer, der Mitglied in der Gruppe ist, erhält dieses PSO als Grundlage für seine Passwortrichtlinien. Für alle weiteren Aktionen, die die PSOs betreffen, gibt es eine umfangreiche Hilfe im `samba-tool`.

15.6 Die Remote Server Administration Tools (RSAT)

Microsoft hat für die Verwaltung einer AD-Domäne Werkzeuge bereitgestellt, mit denen Sie die Domäne von einer Windows-Workstation aus administrieren können. Die Workstation muss Mitglied der Domäne sein, die Sie von dort aus verwalten wollen. In diesem Abschnitt geht es darum, die *Remote Server Administration Tools (RSAT)* zu installieren und dann Gruppen und Benutzer über die RSAT zu verwalten.

15.6.1 Die RSAT einrichten

Um Benutzer und Gruppen über die RSAT verwalten zu können, benötigen Sie einen Windows-10/11-PC, der Mitglied der Domäne ist.



Passen Sie den DNS-Server auf dem Client an

Damit Sie den Client überhaupt erfolgreich in die Domäne aufnehmen können, sorgen Sie dafür, dass auf der Workstation in den Einstellungen der Netzwerkumgebung der DNS-Server der neuen Domäne eingetragen ist. Ohne diesen Eintrag funktioniert der Beitritt zur Domäne nicht, da der Client den Domaincontroller über DNS sucht. Der Client sucht übrigens nicht nur den Namen des Domaincontrollers über DNS, sondern auch die Dienste *Kerberos* und *LDAP*.

Aus diesem Grund nehmen Sie jetzt erst den Windows-Rechner in die Domäne auf. Melden Sie sich hierfür als lokaler Administrator an Ihrer Windows-Workstation an.

15.6.2 Beitritt eines Windows-Clients zur Domäne

Um der Samba4-Domäne beizutreten, öffnen Sie den Explorer und klicken mit der rechten Maustaste auf DIESER PC. Klicken Sie dann auf EIGENSCHAFTEN. Unter Windows 10 wählen Sie den Punkt EINSTELLUNGEN ÄNDERN. Klicken Sie dann auf ÄNDERN. Wählen Sie im nächsten Fenster DOMÄNE aus, und geben Sie Ihren Domänennamen ein. Hier im Buch wäre das example. Geben Sie im folgenden Fenster als Benutzer Administrator und dessen Passwort ein, und klicken Sie auf OK. Unter Windows 11 können Sie direkt nach dem Öffnen der Eigenschaften auf DOMÄNE ODER ARBEITSGRUPPE klicken und der Domäne beitreten.

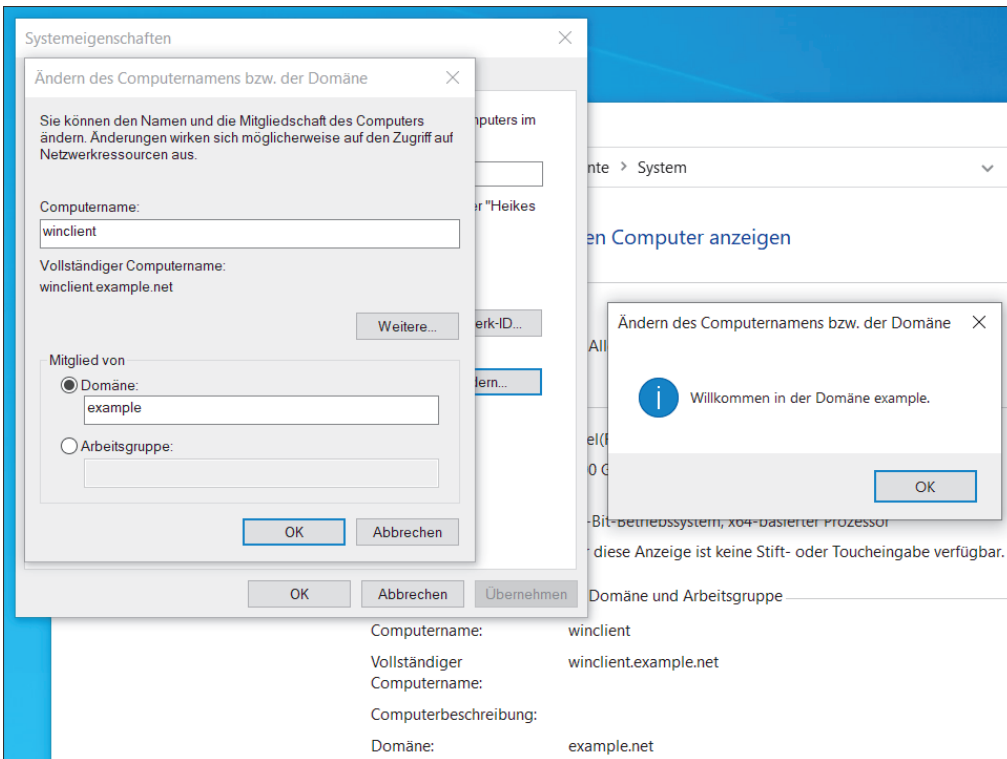


Abbildung 15.1 Einen Client in die Domäne aufnehmen

Nach einem Klick auf die Schaltfläche OK dauert es eine Weile, bis Sie die Meldung WILLKOMMEN IN DER DOMÄNE EXAMPLE erhalten. Um die Einstellung wirksam werden zu lassen, starten Sie Windows neu. Nach dem Neustart können Sie sich jetzt als *Domänenadministrator* anmelden. Denken Sie daran, dass Sie bei der Anmeldung als Domänenadministrator

immer den Namen der Domäne in der Form `example\Administrator` mit angeben, da sonst der lokale Administrator für die Anmeldung verwendet wird. Jeden weiteren Windows-Client nehmen Sie genau so in die Domäne auf. Abbildung 15.1 zeigt die Fenster zur Aufnahme in die Domäne.



Wenn Sie sich als *Administrator* anmelden, stellen Sie immer den Domänennamen voran, da sonst eine Anmeldung als lokaler Administrator durchgeführt wird. Wenn Sie sich als Benutzer der Domäne anmelden, reicht der Benutzername.

15.6.3 Einrichten der RSAT

Um die benötigten RSAT zu aktivieren, starten Sie die Windows-10/11-App *Apps & Features* und fügen die optionalen Features aus Abbildung 15.2 hinzu.

	RSAT: DNS-Servertools	5,87 MB
	RSAT: Server-Manager	29,7 MB
	RSAT: Tools für Active Directory Domain Services und Lightweight Directory Services	16,3 MB
	RSAT: Tools zur Gruppenrichtlinienverwaltung	17,7 MB

Abbildung 15.2 Konfiguration der RSAT

Schließen Sie die App. Anschließend finden Sie im Startmenü unter **WINDOWS-VERWALTUNGSPROGRAMME** alle benötigten Programme zur Verwaltung der Domäne. Aus Platzgründen zeigen wir hier nur die Einrichtung unter Windows 10. Unter Windows 11 heißen die zusätzlich benötigten Features identisch.

15.6.4 Benutzer- und Gruppenverwaltung mit den RSAT

Wenn Sie das Tool *Active Directory-Benutzer und -Computer* starten, können Sie die von Ihnen erstellte Domäne sehen sowie Benutzer und Gruppen verwalten. Wenn Sie einen neuen Benutzer, eine neue Gruppe oder einen neuen Host anlegen wollen, führen Sie einen Rechtsklick auf die rechte Seite des Fensters aus. Dann öffnet sich ein Kontextmenü. Dort klicken Sie auf **NEU**, und es öffnet sich ein neues Menü, in dem Sie dann das entsprechende Objekt auswählen können. In Abbildung 15.3 sehen Sie als Beispiel das Anlegen eines neuen Benutzers. So können Sie jetzt Schritt für Schritt alle Benutzer und Gruppen über Ihren Windows-Client anlegen und verwalten.

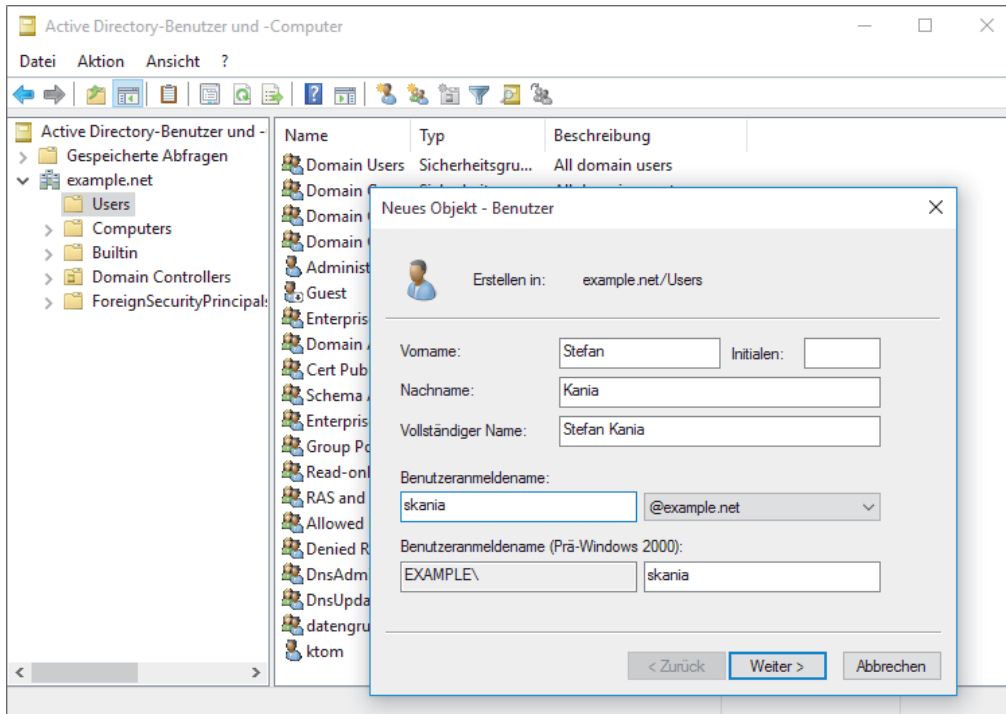


Abbildung 15.3 Anlegen eines neuen Benutzers

Bevor Sie die ersten Gruppen und Benutzer anlegen, sollten Sie sich Gedanken darüber machen, wie Sie Ihre Struktur des Active Directory gestalten wollen. Wenn Sie später Gruppenrichtlinien einsetzen wollen, benötigen Sie eine komplett eigene Struktur mit Organisationseinheiten, da Sie für den Container *Users* keine Gruppenrichtlinien festlegen können.

15.7 Gruppenrichtlinien

In einer Domäne können Sie über Gruppenrichtlinien (GPOs) Berechtigungen für Ressourcen vergeben oder Voreinstellungen für die Benutzer und Hosts vornehmen. Mit Samba 4 können Sie auch GPOs verwalten und erstellen. Mit dem Konsolenwerkzeug *samba-tool* sind Sie in der Lage, die Gruppenrichtlinien eingeschränkt über die Kommandozeile zu verwalten.

15.7.1 Verwaltung der GPOs mit den RSAT

Für die Verwaltung der Gruppenrichtlinien stellen Ihnen die RSAT die Werkzeuge *Gruppenrichtlinienverwaltung* und *Gruppenrichtlinieneditor* zur Verfügung.

15.7.2 Erste Schritte mit der Gruppenrichtlinienverwaltung

Das Werkzeug zur Verwaltung der Gruppenrichtlinien finden Sie im Menü im Unterpunkt WINDOWS-VERWALTUNGSPROGRAMME.

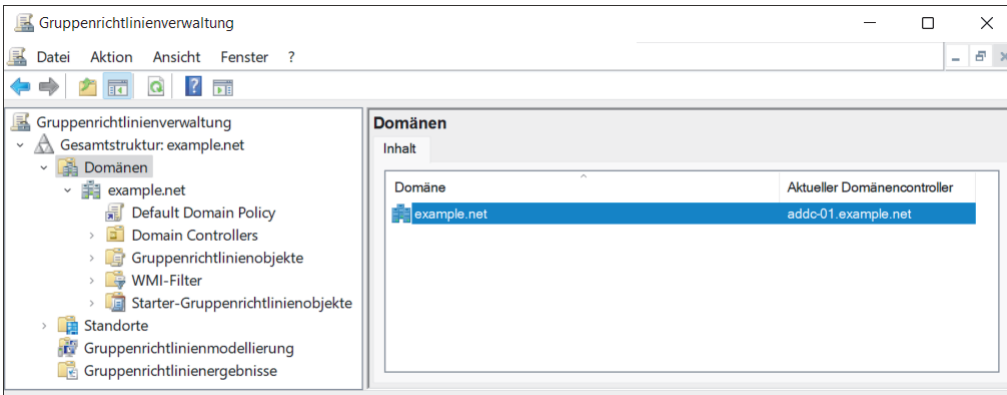


Abbildung 15.4 Die Gruppenrichtlinienverwaltung

In Abbildung 15.4 sehen Sie das Werkzeug für die Verwaltung der Gruppenrichtlinien, ohne dass Änderungen vorgenommen wurden. Wenn Sie hier den Unterpunkt GRUPPENRICHTLINIENOBJEKTE anklicken, sehen Sie alle bereits existierenden Gruppenrichtlinien.

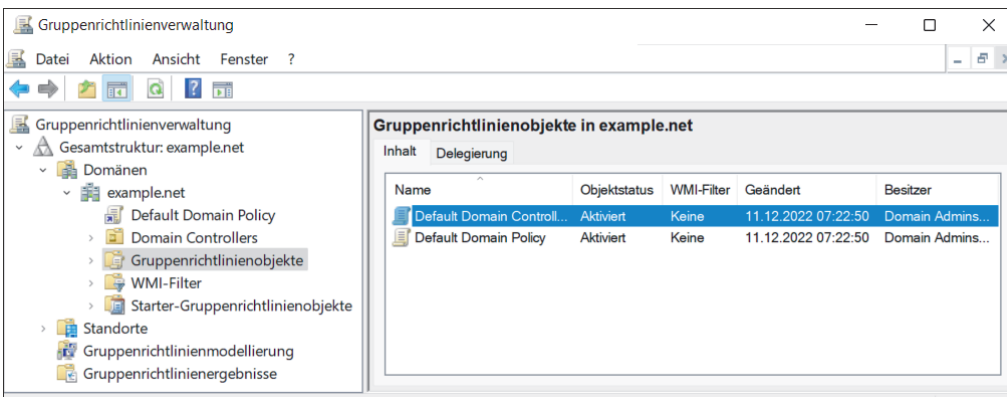


Abbildung 15.5 Die Standardgruppenrichtlinien

In Abbildung 15.5 sehen Sie die beiden Standardgruppenrichtlinien DEFAULT DOMAIN CONTROLLERS POLICY und DEFAULT DOMAIN POLICY.

Wenn Sie jetzt auf eine der bestehenden Gruppenrichtlinien klicken, so erhalten Sie eine Übersicht über die Pfade, mit denen diese Gruppenrichtlinie verknüpft ist, und erfahren, für welche Objekte die Gruppenrichtlinie angewendet wird. In Abbildung 15.6 sehen Sie die

Standardeinstellungen der DEFAULT DOMAIN POLICY. Wie Sie hier sehen, ist diese Gruppenrichtlinie für alle angemeldeten Benutzer der Domäne gültig und ist auch als aktiv markiert. Wollen Sie jetzt eine neue Gruppenrichtlinie erstellen, können Sie diese direkt in dem Container Gruppenrichtlinienobjekte erstellen und später mit den gewünschten OUs verlinken.

Da GPOs nur ab der OU wirksam sind, mit der sie verknüpft sind, benötigen Sie mindestens eine OU, in die Sie anschließend alle Benutzer-, Gruppen- und Computer-Objekte verschieben, für die diese GPO wirksam sein soll.

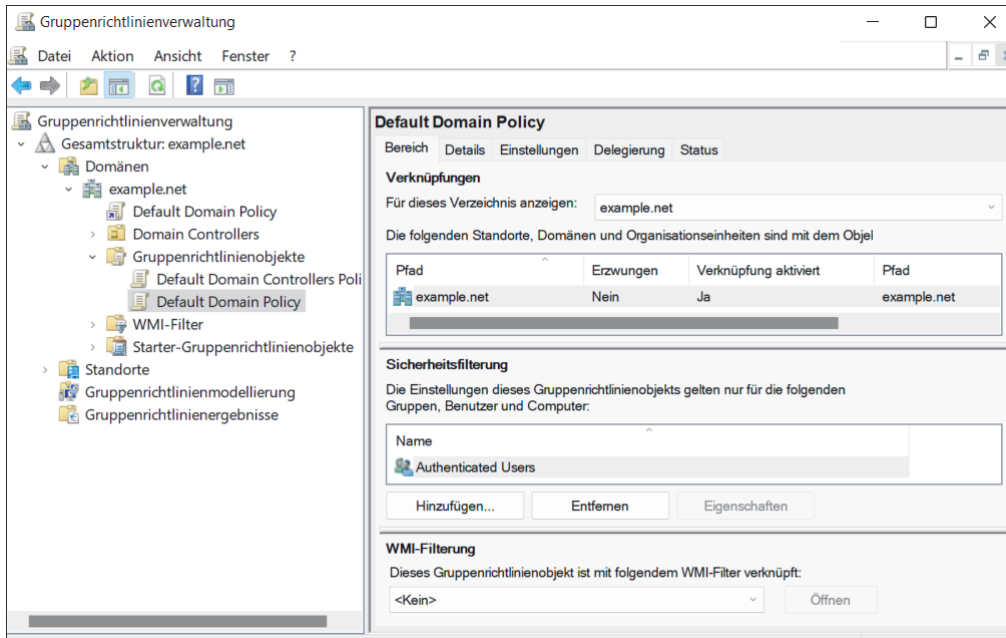


Abbildung 15.6 Standardeinstellungen der »Default Domain Policy«

Gerade wenn Sie sehr viele Gruppenrichtlinien erstellen möchten, ist es aber sinnvoll, die Gruppenrichtlinien auch mit Untercontainern zu verknüpfen, um an verschiedenen Stellen mit unterschiedlichen Rechten oder Einschränkungen arbeiten zu können, da nur dadurch unterschiedliche GPOs für verschiedene Benutzer und Gruppen wirksam werden können.

Aus diesem Grund sollten Sie die Struktur Ihrer Domäne sorgfältig planen, Sie ersparen sich damit eine spätere komplette Überarbeitung.

15.7.3 Eine Gruppenrichtlinie erstellen

Wenn Sie jetzt eine neue GPO erstellen wollen, starten Sie als Domänenadministrator in den RSAT die GRUPPENRICHTLINIENVERWALTUNG. Öffnen Sie die Ansicht auf der linken Seite, bis Sie Ihre Domäne und darunter den Ordner GRUPPENRICHTLINIENOBJEKTE sehen. Klicken Sie

mit der rechten Maustaste auf GRUPPENRICHTLINIENOBJEKTE und anschließend auf NEU. Es erscheint ein neues Fenster, das Sie in Abbildung 15.7 sehen können.

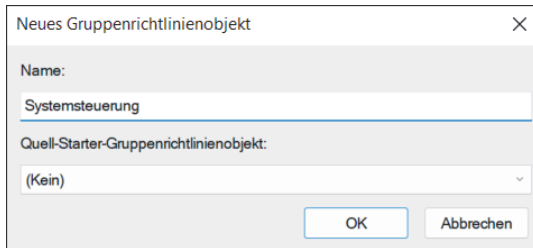


Abbildung 15.7 Einen Namen für die neue GPO vergeben

Vergeben Sie einen Namen für die neue GPO. In diesem Fall soll mit der GPO die Verwendung der Systemsteuerung unterbunden werden.

Bestätigen Sie die Eingabe mit einem Klick auf OK. Anschließend sehen Sie im Container GRUPPENRICHTLINIENOBJEKTE die neue GPO so wie in Abbildung 15.8.

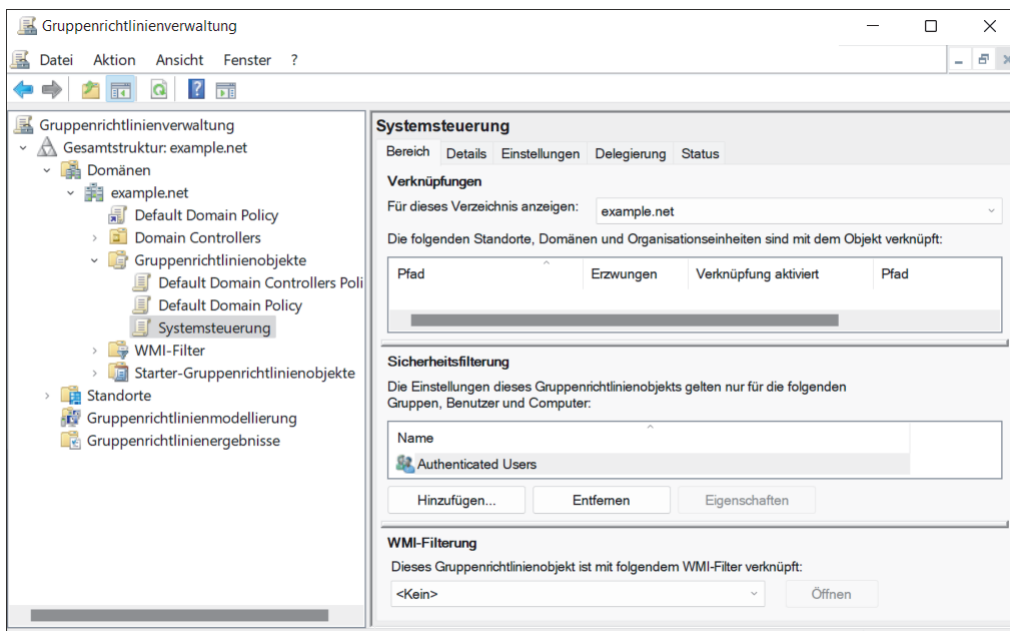


Abbildung 15.8 Die neue GPO

Damit haben Sie eine leere GPO erstellt, in der Sie jetzt die entsprechenden Einstellungen vornehmen. Dazu klicken Sie mit der rechten Maustaste auf die GPO und dann auf BEARBEITEN. Daraufhin öffnet sich der GRUPPENRICHTLINIENVERWALTUNGS-EDITOR.

Dort sehen Sie auf der linken Seite zwei Bereiche: die COMPUTERKONFIGURATION und die BENUTZERKONFIGURATION. Bei der Einschränkung, die hier eingerichtet werden soll, handelt es sich um eine Benutzer-GPO, da die Systemsteuerung unabhängig von der Workstation sein soll, an der sich ein Benutzer anmeldet. Öffnen Sie die Struktur unterhalb von BENUTZERKONFIGURATION • ADMINISTRATIVE VORLAGEN, und klicken Sie dort auf den Ordner SYSTEMSTEUERUNG. Jetzt sehen Sie, wie in Abbildung 15.9 dargestellt, auf der rechten Seite die Möglichkeiten, die Sie haben, um die GPO anzupassen.

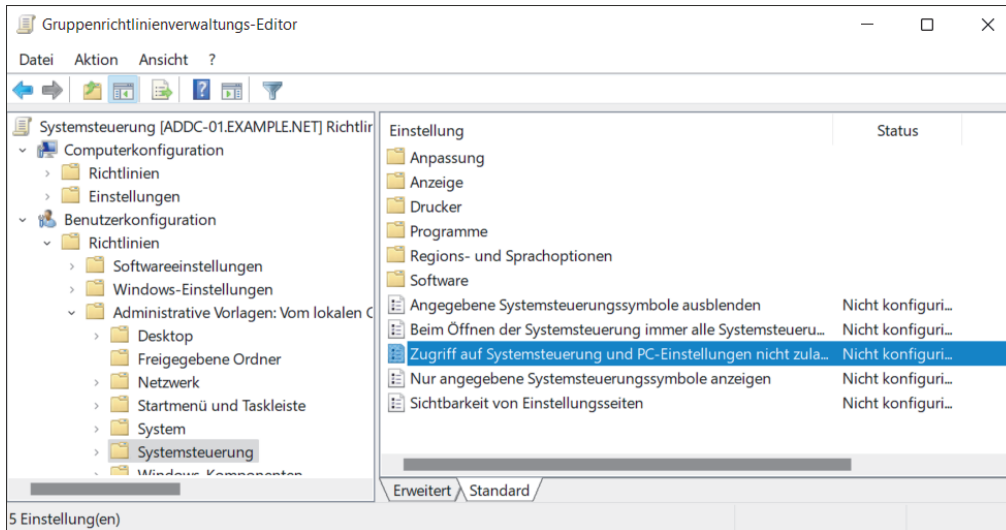


Abbildung 15.9 Anpassen der GPO

Die Richtlinie, die für die GPO benötigt wird, heißt ZUGRIFF AUF DIE SYSTEMSTEUERUNG UND PC-EINSTELLUNGEN NICHT ZULASSEN. Ein Doppelklick auf die Einstellung öffnet ein neues Fenster. In diesem Fenster aktivieren Sie die Einschränkung (siehe Abbildung 15.10).

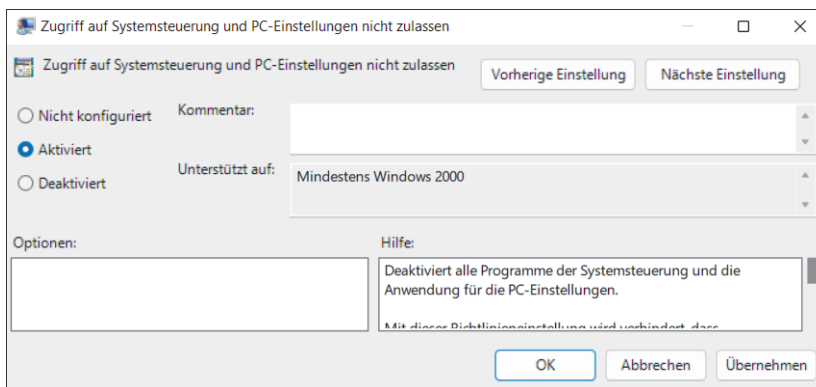


Abbildung 15.10 Aktivierung der Einschränkung

Bestätigen Sie die Einstellung mit einem Klick auf OK. Im Anschluss daran können Sie den GRUPPENRICHTLINIENVERWALTUNGS-EDITOR schließen. Sie gelangen dann wieder in die GRUPPENRICHTLINIENVERWALTUNG. Führen Sie hier einen Doppelklick auf die neue GPO aus, und klicken Sie dann auf den Karteireiter EINSTELLUNGEN und danach auf SHOW ALL. Jetzt sehen Sie, so wie in Abbildung 15.11, alle Einstellungen der GPO.

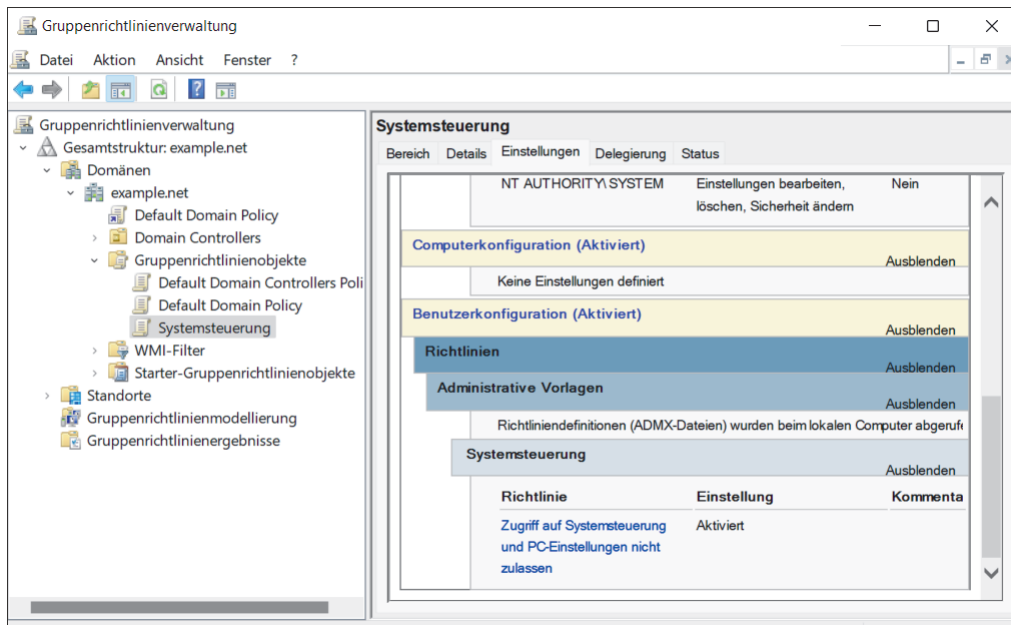


Abbildung 15.11 Einstellungen der neuen Gruppenrichtlinie

15.7.4 Die Gruppenrichtlinie mit einer OU verknüpfen

Bis jetzt haben Sie zwar eine Gruppenrichtlinie (GPO) angelegt, diese aber noch keinem Benutzer oder keiner Gruppe zugewiesen. Gruppenrichtlinien werden in bestimmten für diesen Zweck entworfenen OUs verwaltet und sind dann für Objekte ab dieser OU gültig. Legen Sie im nächsten Schritt eine Struktur für Ihre GPOs an.

Hier ist eine sehr gute Planung vonnöten. Je mehr GPOs Sie in Ihrem Netzwerk einsetzen wollen, desto genauer sollte Ihre Planung sein, da sich Benutzer- und Gruppenobjekte nur in einer OU befinden können.

Im Folgenden soll jetzt eine kleine Struktur angelegt werden, um zu zeigen, wie die GPOs funktionieren. Klicken Sie hierfür mit der rechten Maustaste auf Ihr Domänenobjekt, und wählen Sie den Punkt NEUE ORGANISATIONSEINHEIT aus. Es öffnet sich ein neues Fenster, in dem Sie nur den Namen für die neue OU angeben müssen. Vergeben Sie hier einen Namen, und klicken Sie anschließend auf OK (siehe die Beispiel-OU *Buchhaltung* in Abbildung 15.12).



Aufhebung der Vererbung von GPOs

Einschränkungen, die Sie über Gruppenrichtlinien erstellen, vererben sich immer auf alle Objekte in der entsprechenden Organisationseinheit (OU) und in alle untergeordneten Organisationseinheiten. Es gibt zwar auch hier Möglichkeiten, die Vererbung der Gruppenrichtlinien aufzuheben, dieses Thema sprengt jedoch den Rahmen des Buches.

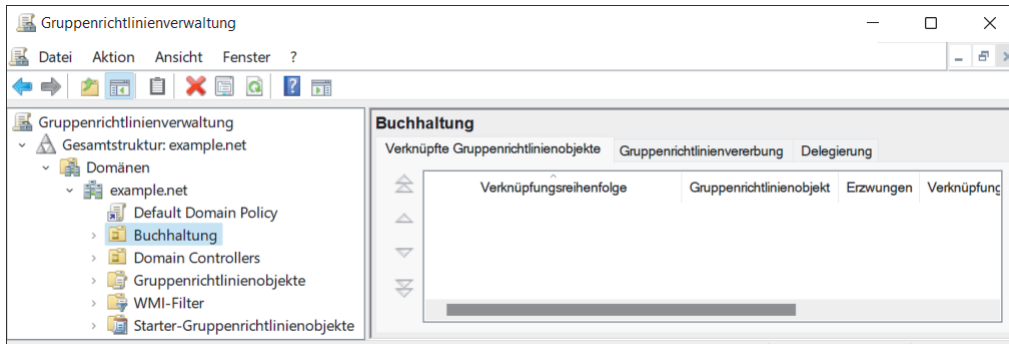


Abbildung 15.12 Anlegen einer neuen OU

15

Wie Sie hier sehen, ist mit dieser OU noch keine GPO verknüpft worden. Im nächsten Schritt soll diese Verknüpfung erstellt werden. Klicken Sie hierfür mit der rechten Maustaste auf die gerade erstellte OU und anschließend auf **VORHANDENES GRUPPENRICHTLINIENOBJEKT VERKNÜPFEN**.

Nun erscheint eine Liste aller vorhandenen GPOs. Doppelklicken Sie hier auf die von Ihnen erstellte GPO. Jetzt sehen Sie, so wie in Abbildung 15.13, die Verknüpfung der GPO mit der OU.

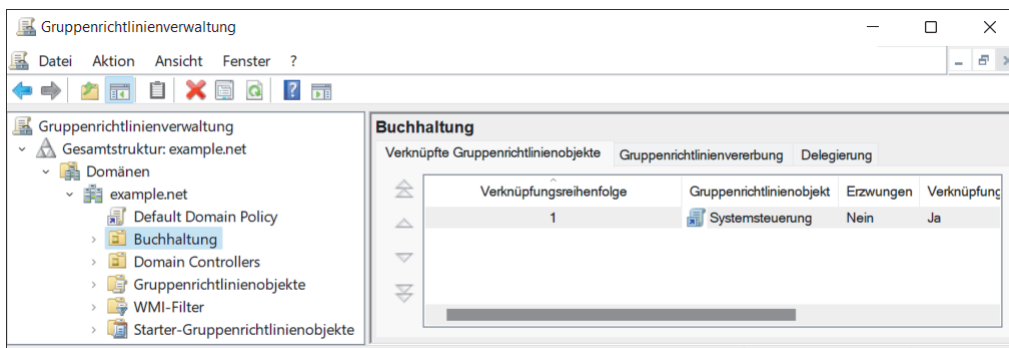


Abbildung 15.13 Eine neue Verknüpfung

Für unser Beispiel haben wir die Gruppe *datengruppe* und den Benutzer *ktom* in die neue OU *Buchhaltung* verschoben und den Benutzer zum Mitglied der Gruppe gemacht. Wech-

seln Sie jetzt auf den Karteireiter **BEREICH**. Im rechten unteren Teil sehen Sie den Punkt **SICHERHEITSFILTERUNG**.

An dieser Stelle ist grundsätzlich immer erst der Eintrag **AUTHENTICATED USERS** vorhanden, sodass diese GPO für alle Benutzer und Gruppen in der OU gültig ist.

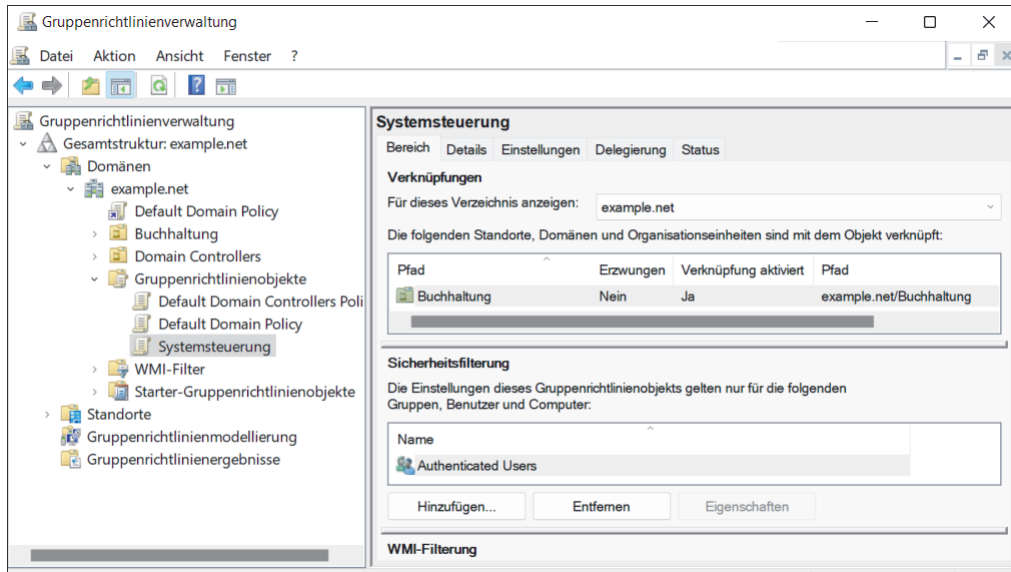


Abbildung 15.14 Die komplette GPO

Hier können Sie jetzt die derzeitigen Einträge löschen und die Gruppe *datengruppe* eintragen, für die diese GPO gelten soll.

Soll die Gruppenrichtlinie für alle Benutzer gelten, können Sie den Eintrag so lassen, wie er ist. Nach allen Anpassungen sieht Ihr Objekt so aus wie in Abbildung 15.14.

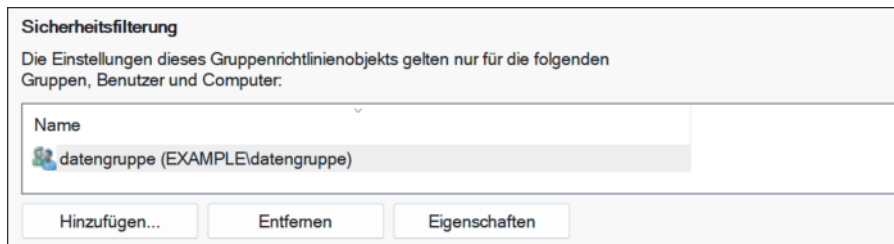


Abbildung 15.15 Alle Objekte in der neuen OU

Wenn Sie das Objekt *Authenticated User* aus der Sicherheitsfilterung entfernen, bekommen Sie eine Meldung angezeigt, die Sie in Abbildung 15.16 sehen können.

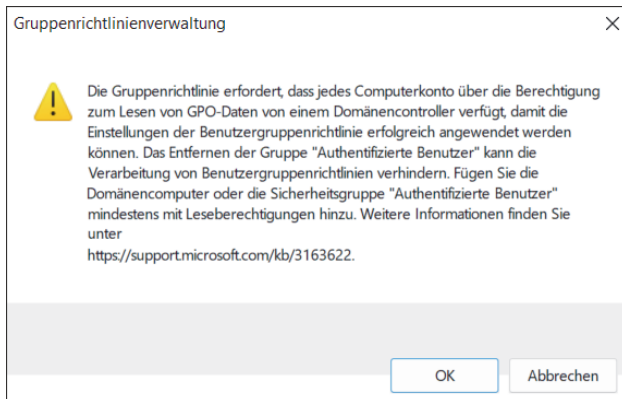


Abbildung 15.16 Warnung zu fehlenden Berechtigungen

Seit einiger Zeit hat Microsoft eine Änderung an den Berechtigungen der GPOs vorgenommen. Wenn Sie jetzt Authenticated Users aus den Sicherheitsfiltern entfernen und eine Gruppe dort eintragen, müssen Sie den Domain Computern das Leserecht an der GPO geben. Die Berechtigung setzen Sie über den Karteireiter DELEGIERUNG. In Abbildung 15.17 sehen Sie die neu gesetzten Berechtigungen.

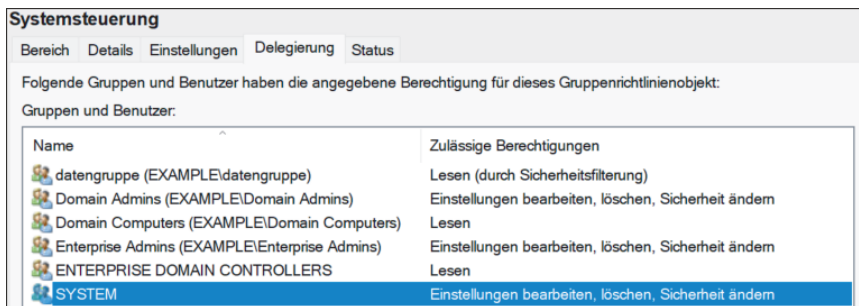


Abbildung 15.17 Rechte für die Domain-PCs

Wenn sich jetzt ein Benutzer aus der Gruppe an einem Client in der Domäne anmeldet, wird er im Startmenü die SYSTEMSTEUERUNG nicht mehr sehen. Ein Rechtsklick auf COMPUTER zeigt nur noch das Fenster aus Abbildung 15.18.

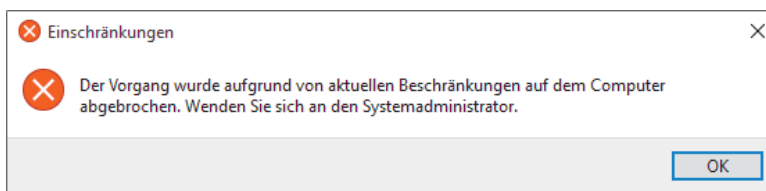


Abbildung 15.18 Warnung auf dem Client

Natürlich ist das nur ein Einstieg in das Thema GPOs. Wie Sie die Gruppenrichtlinien in Ihrer Umgebung organisieren, sollten Sie sorgfältig prüfen, damit die Regeln möglichst übersichtlich und einfach zu verwalten sind.

15.7.5 GPOs über die Kommandozeile

Um die GPOs über die Shell zu verwalten, lassen Sie sich noch einmal alle GPOs anzeigen. Das Folgende ist nur ein Überblick über einige der Möglichkeiten, die Sie mit dem Kommando `samba-tool gpo` haben. Wenn Sie mehr über die Verwaltung der Gruppenrichtlinien mithilfe der Kommandozeile wissen möchten, empfehlen wir Ihnen die Hilfe von `samba-tool`. Sie werden aber sehr schnell feststellen, dass die Verwaltung über die grafische Oberfläche schneller, einfacher und übersichtlicher ist. In Listing 15.42 sehen Sie die Übersicht über alle GPOs, die momentan in der Domäne existieren:

```
root@addc-01:~# samba-tool gpo listall
GPO          : 31B2F340-016D-11D2-945F-00C04FB984F9
display name : Default Domain Policy
path         : \\example.net\sysvol\example.net\Policies\
              31B2F340-016D-11D2-945F-00C04FB984F9
dn          : CN=31B2F340-016D-11D2-945F-00C04FB984F9,\
              CN=Policies,CN=System,DC=example,DC=net
version     : 0
flags       : NONE

GPO          : 25612555-3CC1-4D82-BBBF-E3E2DB54E9B4
display name : Systemsteuerung
path         : \\example.net\SysVol\example.net\Policies\
              25612555-3CC1-4D82-BBBF-E3E2DB54E9B4
dn          : CN=25612555-3CC1-4D82-BBBF-E3E2DB54E9B4,CN=Policies,\
              CN=System,DC=example,DC=net
version     : 65536
flags       : NONE

GPO          : 6AC1786C-016F-11D2-945F-00C04FB984F9
display name : Default Domain Controllers Policy
path         : \\example.net\sysvol\example.net\Policies\
              6AC1786C-016F-11D2-945F-00C04FB984F9
dn          : CN=6AC1786C-016F-11D2-945F-00C04FB984F9,CN=Policies,\
              CN=System,DC=example,DC=net
version     : 0
flags       : NONE
```

Listing 15.42 Übersicht über alle GPOs

Hier sehen Sie, dass jetzt neben den beiden Standard-GPOs auch noch die zuvor angelegte GPO sichtbar ist. GPOs werden als Verzeichnisse im Dateisystem abgelegt. Die Verzeichnisse für die Gruppenrichtlinien finden Sie unter `/var/lib/samba/sysvol/example.net/Policies`. Bei dem Verzeichnis `sysvol` handelt es sich um die gleichnamige Freigabe, die später auf alle DCs der Domäne repliziert wird.

Natürlich können Sie sich auch anzeigen lassen, welche Gruppenrichtlinien für einen bestimmten Benutzer wirksam sind, siehe Listing 15.43:

```
root@addc-01:~# samba-tool gpo list ktom
  Systemsteuerung 25612555-3CC1-4D82-BBBF-E3E2DB54E9B4
  Default Domain Policy 31B2F340-016D-11D2-945F-00C04FB984F9
```

Listing 15.43 Auflisten der GPOs für einen bestimmten Benutzer

Die GPO Systemsteuerung wurde ja im letzten Abschnitt der Gruppe Buchhaltung zugewiesen. Der Benutzer `ktom` ist Mitglied dieser Gruppe `datengruppe`, und sein Benutzerobjekt liegt in der Organisationseinheit, mit der die Gruppenrichtlinie verknüpft wurde. Somit ist die GPO auch für ihn relevant. Alle OUs, auf die diese GPO verlinkt wurde, können Sie sich so wie in Listing 15.44 auflisten lassen:

```
root@addc-01:~# samba-tool gpo listcontainers \
  25612555-3CC1-4D82-BBBF-E3E2DB54E9B4
Container(s) using GPO 25612555-3CC1-4D82-BBBF-E3E2DB54E9B4
  DN: OU=Buchhaltung,DC=example,DC=net
```

Listing 15.44 Auflistung aller Verknüpfungen einer GPO

Leider können Sie an dieser Stelle nicht den Namen der GPO verwenden, sondern müssen den eindeutigen Schlüssel benutzen. Sie haben auf diese Weise zwar auch die Möglichkeit, GPOs zu erstellen, zu löschen und zu verändern; diese Aufgaben sind jedoch viel einfacher über die RSAT zu realisieren.

15.8 Linux-Clients in der Domäne

Ein Linux-Client lässt sich auch in eine AD-Domäne einbinden – genau wie ein Windows-Client. In diesem Abschnitt geht es darum, einen Linux-Client, egal ob mit oder ohne GUI, in die Domäne aufzunehmen. Den Anfang werden wir mit der Grundkonfiguration beschreiben, am Ende beschreiben wir dann die Umstellung für einen Linux-Client mit grafischer Oberfläche. Für den Zugriff auf das AD wird der Dienst `winbind` benötigt, zusammen mit PAM wird dann die Authentifizierung durchgeführt. Für die Authentifizierung über PAM installieren Sie lediglich das entsprechenden PAM-Module Ihrer Distribution. Die Einträge zur Konfiguration in den entsprechenden PAM-Dateien werden automatisch während der Installation erzeugt.

Als Linux-Client können Sie jede der hier im Buch aufgeführten Distributionen verwenden. (Auch die meisten anderen hier nicht besprochenen Distributionen bringen Samba-4-Pakete mit und können verwendet werden.) Allerdings ist die Art und Weise, wie Sie die Client-Pakete installieren, von Distribution zu Distribution unterschiedlich. Sind die Pakete einmal installiert, ist die Konfiguration immer identisch. Achten Sie bei der Installation immer darauf, dass Sie neben den Samba-Paketen ebenfalls die Winbind-Pakete installieren.



Welche Pakete für den Samba-Client?

Ob Sie für einen Linux-Client eine eigene Quelle für die Samba-Pakete nutzen oder die Pakete aus dem Repository der Distribution verwenden, spielt hier keine große Rolle, da die Funktionalität der Dateidienste bei allen gegeben ist. Nur bei Domaincontrollern oder CTDB-Servern ist es sinnvoll, immer die aktuellste Version zu nutzen.

Bei der Installation der Pakete auf dem Client können Sie die Paketliste des Domaincontrollers übernehmen. Die Pakete *bind9* und *dnsutils* werden auf einem Client nicht benötigt.

Welche Distribution Sie verwenden wollen, hängt wieder davon ab, was für eine Art Client Sie installieren möchten. Bei einem reinen Client spielt die verwendete Samba-Version keine Rolle: Alle von den aktuellen Distributionen eingesetzten Samba-Versionen unterstützen das SMB-Protokoll in der Version 3.x.

Unter openSUSE installieren Sie die Samba-Client-Pakete über YaST. Wenn Sie die Standardinstallation durchgeführt haben, sind die Samba-Pakete bereits auf dem System vorhanden. Wenn Sie die Kerberos-Authentifizierung auf der Kommandozeile benötigen, installieren Sie auf jeden Fall zusätzlich das Paket *krb5-user*. Die Konfiguration des Clients wird immer über die Datei */etc/samba/smb.conf* durchgeführt. Nach der Installation der Pakete ist bereits eine *smb.conf* vorhanden. Diese Datei ist aber in den seltensten Fällen direkt nutzbar. Wir werden hier mit einer komplett leeren Datei beginnen und nur die wirklich notwendigen Parameter verwenden. Halten Sie die Konfigurationsdatei immer so klein wie möglich, das gilt besonders für Fileserver. Bei jeder Änderung der Datei wird immer die gesamte Datei an alle Clients übertragen, die eine Verbindung zum Fileserver haben. Um auf dem Linux-Client auch die Kerberos-Authentifizierung durchführen zu können, erstellen Sie im Anschluss an die Installation die Datei */etc/krb5.conf*. Da die Datei auf allen Ihren Clients identisch ist, können Sie die Datei später einfach auf alle Clients kopieren. In Listing 15.45 sehen Sie den Inhalt der Datei:

```
[libdefaults]
    default_realm = EXAMPLE.NET
    dns_lookup_realm = false
    dns_lookup_kdc = true
```

Listing 15.45 »krb5.conf« für die Clients

**Achten Sie auf den DNS-Server**

Achten Sie darauf, dass der Domaincontroller Ihrer Domäne als DNS-Server auf allen Clients eingetragen ist. Alle Informationen zur Domäne bekommt ein Client immer vom DNS-Server der Active Directory-Domäne.

In der Datei `/etc/samba/smb.conf` werden sowohl der `winbind` als auch der `smbd` und der `nmbd` konfiguriert. Dazu erstellen Sie eine neue `/etc/samba/smb.conf` mit dem Inhalt aus Listing 15.46:

```
[global]
    workgroup = example
    realm = EXAMPLE.NET
    security = ADS
    winbind refresh tickets = yes
    winbind use default domain = yes
    template shell = /bin/bash
    idmap config * : range = 100000 - 199999
    idmap config EXAMPLE : backend = rid
    idmap config EXAMPLE : range = 1000000 - 1999999
```

15

Listing 15.46 Konfiguration des »winbind«

Je nach Distribution ist bereits eine `/etc/samba/smb.conf` vorhanden. Sie können die Datei, die von der Distribution mitgeliefert wird, einfach löschen und durch die Einträge aus Listing 15.46 ersetzen. Die Parameter haben die folgenden Bedeutungen:



- ▶ `workgroup = example`
Hier wird der NetBIOS-Name der Domäne angegeben. Auch als Mitglied im AD heißt der Parameter `workgroup`.
- ▶ `realm = EXAMPLE.NET`
Bei dem `realm` handelt es sich um die Information für die Kerberos-Domäne. Für diesen Realm wird sich der Samba-Server einen KDC suchen.
- ▶ `security = ADS`
Damit legen Sie fest, dass Ihr Server ein Mitglied in einer AD-Domäne ist.
- ▶ `winbind refresh tickets = yes`
Mit diesem Parameter werden Kerberos-Tickets automatisch erneuert, wenn der Benutzer angemeldet ist und das Ticket abläuft.
- ▶ `winbind use default domain = yes`
Haben Sie nur eine Domäne, können Sie mit diesem Parameter dafür sorgen, dass nur die Benutzernamen von `winbind` übergeben werden, ohne die Domäne vor den Namen zu stellen.

- ▶ `template shell = /bin/bash`
Diesen Parameter dürfen Sie auf gar keinen Fall vergessen. Ohne ihn kann sich ein Benutzer aus dem AD zwar anmelden, aber er wird sofort wieder abgemeldet, da der Benutzer im AD keine Shell zugewiesen bekommt.
- ▶ `idmap config * : range = 100000 - 199999`
Neben den Gruppen und Benutzern, die Sie als Administrator anlegen, gibt es noch die *Builtin-Gruppen*. Diese Gruppen haben einen eigenen verkürzten SID. Diese Gruppen benötigen auch das ID-Mapping. Die Konfiguration der *Buidin-Gruppen* wird über den Stern (* eingeleitet. Eigentlich müssten Sie auch noch den Parameter `idmap config * : backend = tdb` konfigurieren, aber dieser Parameter wird von Samba4 automatisch gesetzt. Testen können Sie das mit dem Kommando `testparm`.
- ▶ `idmap config EXAMPLE : backend = rid`
Die IDs der Benutzer werden aus den SIDs der AD-Benutzer generiert. Dazu gibt es verschiedene Möglichkeiten. Die Standardeinstellung für den *winbind* ist die Verwendung von *tdb*-Dateien. Dabei werden zufällige UIDs generiert, den Benutzern zugewiesen und in der *tdb*-Datei gespeichert. Der Nachteil dieses Verfahrens ist, dass so jeder Benutzer auf jedem Linux-System eine andere UID bekommt. Durch den Wechsel auf das Backend `idmap_rid` wird immer der RID des Benutzers aus der AD-Domäne verwendet, um eine eindeutige ID zu generieren. Da dieser eindeutig ist, ist die ID der Benutzer und Gruppen auf allen Linux-Systemen immer eindeutig. Der Benutzer hat dadurch auf allen Linux-Systemen in der gesamten Domäne immer dieselbe UID.

Ein Problem bekommen Sie so aber nicht in den Griff: Auf den DCs Ihrer Domäne werden die UIDs immer im AD verwaltet und somit immer anders als auf den anderen Systemen in der Domäne. Die einfachste Möglichkeit, dieses Problem zu umgehen, besteht darin, die Domaincontroller nicht als Fileserver zu verwenden und alle Dateien immer auf anderen Samba-Servern in der Domäne abzulegen.
- ▶ `idmap config EXAMPLE : = 1000000 - 1999999`
Hier legen Sie den Bereich fest, in dem sich die UIDs der Benutzer befinden sollen.



Das Backend »rid«

Wenn Sie das Backend `rid` verwenden, ist der *winbind* für das ID-Mapping zwingend vorgeschrieben. Eine Nutzung des *sssd* ist nicht mehr möglich. Das betrifft alle Samba-Versionen seit 4.8. Nur im Zusammenhang mit dem Backend *ad* ist die Verwendung des *sssd* noch möglich. Die Verwendung des Backends *ad* ist nicht Bestandteil dieses Buches.

Nachdem Sie die Pakete installiert, die *smb.conf* konfiguriert und die *krb5.conf* kopiert haben, können Sie jetzt die Verbindung zum Domaincontroller testen, indem Sie mit `kinit` ein Ticket für den Administrator der Domäne beziehen. Listing 15.47 zeigt diesen Vorgang:

```

root@client-01:~# kinit administrator
administrator@EXAMPLE.NET's Password:
root@fs-01:~# klist
Credentials cache: FILE:/tmp/krb5cc_0
Principal: administrator@EXAMPLE.NET

```

```

      Issued                Expires                Principal
Dec 22 13:03:15 2022  Dec 22 23:03:15 2022  krbtgt/EXAMPLE.NET@EXAMPLE.NET

```

Listing 15.47 Beziehen eines Kerberos-Tickets

Jetzt sollten Sie auf jeden Fall noch einen Blick in die Datei `/etc/hosts` werfen und sicherstellen, dass ihr Inhalt dem aus Listing 15.48 entspricht:

```

127.0.0.1      localhost
192.168.56.91 client-01.example.net client-01

```

Listing 15.48 Inhalt der Datei `»/etc/hosts«`

Wenn der Client dort nicht mit seiner IP-Adresse eingetragen ist, kommt es zu einer Fehlermeldung hinsichtlich des Namens beim nachfolgenden *Join*.

Jetzt können Sie mit dem Linux-Client der Domäne beitreten. In Listing 15.49 sehen Sie, wie Sie über die Kommandozeile den Linux-Client in die AD-Domäne bringen:

```

root@client-01:~# net ads join -Uadministrator
Enter administrator's password:
Using short domain name -- EXAMPLE
Joined 'CLIENT-01' to dns domain 'example.net'

```

Listing 15.49 Beitritt eines Linux-Clients

Wenn Sie im Anschluss oder später den Beitritt zur Domäne prüfen wollen, können Sie das so wie in Listing 15.50 durchführen:

```

root@client-01:~# net ads testjoin
Join is OK

```

Listing 15.50 Testen des Beitritts

Jetzt können Sie die Samba-Dienste und den *winbind* das erste Mal von Hand starten. Wenn Sie Debian als Client einsetzen, sind die Dienste bereits gestartet. Starten Sie die Dienste dann neu. Wenn Sie openSUSE als Client einsetzen, aktivieren und starten Sie den *winbind*-, den *smb*- und den *nmbd*-Dienst über YaST.

Führen Sie jetzt am besten einen Neustart des Systems durch, um sicher zu sein, dass die Dienste auch bei einem Neustart alle automatisch gestartet werden.



Stoppen Sie den »nscd«

Gerade bei openSUSE wird der *NameService Caching Daemon (nscd)* zur Zwischenspeicherung der Anmeldeinformationen gestartet. Da der *nscd* sich nicht mit dem *winbindd* verträgt, sollten Sie den *nscd* auf jeden Fall deaktivieren oder deinstallieren.

Mit dem Kommando `wbinfo -u` sehen Sie jetzt alle Benutzer aus der AD-Domäne. Mit dem Kommando `wbinfo -g` sehen Sie alle Gruppen aus der AD-Domäne (siehe Listing 15.51):

```
root@client-01:~# wbinfo -u
administrator
dns-addc-01
krbtgt
guest
ktom

root@client-01:~# wbinfo -g
domain users
schema admins
domain guests
domain computers
dnsupdateproxy
group policy creator owners
dnsadmins
allowed rodc password replication group
ras and ias servers
enterprise admins
domain controllers
denied rodc password replication group
cert publishers
domain admins
read-only domain controllers
datengruppe
protected users
enterprise read-only domain controllers
```

Listing 15.51 Auflistung aller Benutzer und Gruppen mit »wbinfo«

Jetzt machen Sie die Benutzer und Gruppen noch im Linux-System bekannt. Um die Benutzer aus verschiedenen Authentifizierungsquellen auslesen zu können, verwendet Linux den *Name Service Switch (NSS)*. Dieser wird über die Datei `/etc/nsswitch.conf` konfiguriert. Damit Ihr System die von *winbind* übergebenen Benutzer auch im Linux-System verwenden kann, passen Sie die Datei `/etc/nsswitch.conf` so wie in Listing 15.52 an:

```
passwd compat systemd winbind
group  compat systemd winbind
```


Listing 15.52 Die Datei »/etc/nsswitch.conf«

Wenn Sie jetzt die Kommandos `getent passwd <username>` und `getent group <groupname>` verwenden, werden Ihnen die Benutzer und Gruppen aus der AD-Domäne als Linux-Benutzer angezeigt. In Listing 15.53 sehen Sie das Ergebnis einer Benutzerabfrage:


```
root@client-01:~# getent passwd ktom
ktom:*:1001107:1000513::/home/EXAMPLE/ktom:/bin/bash
```

Listing 15.53 Ergebnis von »`getent passwd <username>`«

Wenn Sie die UIDs der Benutzer jetzt mit den RIDs der AD-Benutzer vergleichen, sehen Sie, dass beide identisch sind. Um sich den SID eines Benutzers anzeigen zu lassen, verwenden Sie das Kommando `wbinfo -n ktom`. Das Kommando zeigt den vollständigen SID des Benutzers. Bei dem letzten Teil des SID handelt sich um den RID, der dazu verwendet wird, die UID zu erzeugen. Wenn Sie den Wert mit der UID des Ergebnisses von `getent passwd ktom` vergleichen, sehen Sie, dass der RID zum Anfangswert Ihrer ID-Mapping-Range aufaddiert wird. Jetzt ist der Linux-Client so konfiguriert, dass die Benutzer aus der AD-Domäne sich über `ssh` auch an den Linux-Clients anmelden können. Da die Freigaben noch nicht eingebunden sind, wird der Benutzer zu diesem Zeitpunkt noch ohne Heimatverzeichnisse angemeldet.

Wenn Sie an dieser Stelle die Anmeldung eines Benutzers – entweder direkt oder über `ssh` – testen, werden Sie mit dem Kommando `klist` feststellen, dass der Benutzer auch sofort ein *TGT* vom Kerberos erhalten hat. Das ist wichtig, da Sie nur so später ein *Single Sign-on* auf andere Dienste realisieren können. 

Für die Heimatverzeichnisse der Benutzer haben Sie jetzt zwei Möglichkeiten. Entweder Sie richten für die Heimatverzeichnisse und andere Datenverzeichnisse einen NFS-Server ein und stellen die Daten über diesen zur Verfügung, oder Sie nutzen `cifs-mount` für die Freigaben. Wie Sie einen NFS-Server einrichten können, lesen Sie in Kapitel 16, »NFS«.

Das System wird das Heimatverzeichnis des Benutzers standardmäßig immer im Verzeichnis `/home/<Domainname>` suchen. Der Grund dafür ist der Standardwert des Parameters `template homedir = /home/%D/%U`, wobei `%D` für den Domainnamen und `%U` für den Benutzernamen stehen. 

15.8.1 Bereitstellen von Freigaben

Wir haben uns entschieden, die gesamte Client-Konfiguration in einen Abschnitt zu schreiben. Dazu gehört auch die Einrichtung des Zugriffs auf die Freigaben eines Servers. Der Server ist aber zurzeit noch nicht konfiguriert, deshalb sollten Sie mit den folgenden Schritten bis nach dem Abschnitt 15.9 »Zusätzliche Server in der Domäne« warten in dem der erste

Fileserver eingerichtet wird. Aber so haben Sie für die Zukunft alle Schritte zur Einrichtung eines Linux-Clients an einer Stelle. Damit Sie auf den Linux-Clients auch Freigaben via CIFS mounten können, ist noch etwas Konfigurationsaufwand notwendig. Die Freigaben sollen ja automatisch gemountet werden und die Authentifizierung am Fileserver soll via Kerberos durchgeführt werden. Dafür werden wir *pam-mount* verwenden.

15.8.2 Mounten über »pam_mount«

Die einfachste Möglichkeit, die Freigaben eines Samba-Servers mittels *cifs* zu mounten, ist die über *pam_mount*. Die Vorbereitung ist hier wieder für die verschiedenen Distributionen jeweils anders. Deshalb gehen wir in den nächsten beiden Abschnitten auf die Vorbereitung getrennt ein.

Vorbereitung unter Debian und Ubuntu

Damit Sie *pam_mount* verwenden können, installieren Sie unter Debian und Ubuntu als Erstes das Paket *libpam-mount*. Bei der Installation des Pakets werden die beiden Dateien */etc/pam.d/common-auth* und */etc/pam.d/common-session* um das PAM-Modul *pam_mount.so* erweitert. Dadurch kann beim Zugriff auf die Freigaben sofort die Authentifizierung des zugreifenden Benutzers durchgeführt werden. Die Authentifizierung können Sie dabei über Kerberos realisieren.

Vorbereitung unter openSUSE

Auch bei openSUSE installieren Sie das Paket *pam_mount*. Achten Sie nur darauf, dass das Paket *krb5-client* installiert ist. Da Sie die Authentifizierung an dieser Stelle schon eingerichtet haben, sollte das Paket bereits installiert sein. Leider müssen Sie bei openSUSE immer die PAM-Konfiguration von Hand vornehmen. Denn im Gegensatz zu Debian und Ubuntu werden die entsprechenden Einträge bei der Installation der Pakete nicht automatisch erstellt.



Halten Sie immer eine »root-Konsole« geöffnet

Wenn Sie an den PAM-Konfigurationsdateien arbeiten, raten wir dazu, dass Sie immer eine *root-Konsole* geöffnet halten, denn Änderungen an den Dateien sind sofort wirksam. Wenn Sie jetzt keine *root-Konsole* mehr geöffnet haben, kann es passieren, dass sich niemand, auch nicht der *root*, am System anmelden kann.

Damit das PAM-Modul vom System verwendet wird, müssen Sie zwei Dateien anpassen:

► die Datei »common-auth«

Passen Sie den Inhalt der Datei so wie in Listing 15.54 an:

```

auth    required    pam_env.so
auth    optional    pam_gnome_keyring.so
auth    sufficient  pam_unix.so      try_first_pass
auth    required    pam_winbind.so  use_first_pass
auth    required    pam_mount.so

```

Listing 15.54 Die Datei »common-auth«

► **die Datei »common-session«**

Passen Sie den Inhalt der Datei so wie in Listing 15.55 an:


```

session required    pam_limits.so
session required    pam_unix.so      try_first_pass
session required    pam_winbind.so
session optional    pam_umask.so
session optional    pam_systemd.so
session optional    pam_gnome_keyring.so  \
                    auto_start only_if=gdms, gdm-password, lxdm, lightdm
session optional    pam_env.so
session optional    pam_mount.so

```

Listing 15.55 Die Datei »common-session«

15

Sollte in der Datei *common-session* das Modul *pam_mkhome.so* eingetragen sein, müssen Sie dieses entfernen. Achten Sie bei den Einträgen auf die Reihenfolge. 

Jetzt ist das PAM-System für die Verwendung von *pam_mount* vorbereitet, und Sie können mit der eigentlichen Konfiguration des Moduls beginnen.

Anpassen der Konfigurationsdatei bei allen Distributionen

Über die Datei */etc/security/pam_mount.conf.xml* realisieren Sie die Konfiguration von *pam_mount*.

In Listing 15.56 sehen Sie zwei Einträge: einmal einen Eintrag für ein Datenverzeichnis und einen weiteren Eintrag für die Heimatverzeichnisse der Benutzer, denn über *pam_mount* lassen sich jetzt auch die Heimatverzeichnisse der Benutzer ohne Probleme automatisch bei der Anmeldung mounten.

```

<volume
  fstype="cifs"
  server="fs-01.example.net"
  path="users/$(DOMAIN_USER)"
  mountpoint="/home/EXAMPLE/$(DOMAIN_USER)"
  sgrp="domain users"
  options="sec=krb5,cuid=$(USERID),workgroup=EXAMPLE,vers=3.1.1" />

```

```
<volume
  fstype="cifs"
  server="fs-01.example.net"
  path="abteilungen"
  mountpoint="/abteilungen"
  sgrp="domain users"
  options="sec=krb5,cuid=%(USERID),workgroup=EXAMPLE,vers=3.1.1" />
```

Listing 15.56 Einträge in der Datei »/etc/security/pam_mount.conf.xml«

Die Parameter haben dabei die folgenden Bedeutungen:

- ▶ `user="%(DOMAIN_USER)"`
Durch den Parameter `user` beschränken Sie die Verfügbarkeit der Freigabe auf bestimmte Benutzer – in diesem Fall auf alle Benutzer der Domäne.
- ▶ `fstype="cifs"`
Hier wird der Dateisystemtyp festgelegt – in diesem Fall `cifs`.
- ▶ `server="fs-01.example.net"`
Angabe des Servers auf dem Sie die Freigabe eingerichtet haben.
- ▶ `path="users\\%(DOMAIN_USER)"`
Hierbei handelt es sich um die Freigabe und den Pfad innerhalb der Freigabe, der gemountet werden soll. Im ersten Beispiel wird immer das Heimatverzeichnis des entsprechenden Benutzers gemountet. Im zweiten Beispiel wird nur der Name der Freigabe angegeben, da hier die gesamte Freigabe gemountet werden soll.
- ▶ `mountpoint=/home/EXAMPLE/%(DOMAIN_USER)"`
Der Mountpoint auf dem lokalen System, in den die Freigabe gemountet werden soll.
- ▶ `sgrp="domain users"`
Der Parameter gibt an, in welcher Gruppe der Benutzer Mitglied sein muss, damit diese Mountoption abgearbeitet wird. Hier haben wir die Gruppe `domain users` angegeben. So wird der Mountpoint für jeden im Active Directory existenten Benutzer angelegt, nicht aber für lokale Benutzer die nur auf dem Client vorhanden sind. So sorgen Sie dafür, dass es beim Anmelden von lokalen Benutzern nicht zu Fehlermeldungen kommt.
- ▶ `option="sec=krb5,cuid=%(USERUID),workgroup=EXAMPLE",vers=3.1.1`
Bei dem Parameter `option` können Sie verschiedene Mountparameter angeben. Über die Option `sec=krb5` legen Sie die Sicherheitsstufe bei der Authentifizierung fest (standardmäßig `ntlmv1`). Da Samba 4 Kerberos bereitstellt, soll hier auch Kerberos für die Authentifizierung verwendet werden. Deshalb wird an dieser Stelle der Parameter `sec=krb5` gesetzt. Zusätzlich wird der Parameter `cuid=%(USERUID)` benötigt. Dieser Parameter sorgt dafür, dass der richtige Benutzer-Realm bei der Kerberos-Authentifizierung übergeben wird. Ohne diesen Parameter schlägt die Kerberos-Authentifizierung fehl. Über die Option `workgroup=EXAMPLE` setzen Sie die Domäne, in der sich der Benutzer befindet.



Vergessen Sie auf gar keinen Fall den Parameter `vers=3.1.1`. Dieser Parameter legt fest, welche SMB-Version für den Zugriff auf die Freigabe verwendet wird. Der Standardwert ist hier `SMBv1`, diese Version ist aber unsicher und wird von Samba, in der Grundinstallation auch nicht mehr unterstützt.

Wenn sich jetzt ein Benutzer am Client anmeldet, werden die Freigaben automatisch gemountet und dem Benutzer zugeordnet. Listing 15.57 zeigt die gemounteten Freigaben:

```
//fs-01.example.net/users/ktom on /home/EXAMPLE/ktom type cifs\
(rw,nosuid,nodev,relatime,vers=3.1.1,sec=krb5,cuid=1001107,cache=strict\
,username=ktom,domain=EXAMPLE,uid=1001107,noforceuid,gid=1000513,\
noforcegid,addr=192.168.56.91,file_mode=0755,dir_mode=0755,soft,\
nounix,serverino,mapposix,rsize=4194304,wsize=4194304,bsize=1048576,\
echo_interval=60,actimeo=1,user=ktom)

//fs-01.example.net/abteilungen on /abteilungen type cifs\
(rw,nosuid,nodev,relatime,vers=3.1.1,sec=krb5,cuid=1001107,cache=strict\
,username=ktom,domain=EXAMPLE,uid=1001107,noforceuid,gid=1000513,\
noforcegid,addr=192.168.56.91,file_mode=0755,dir_mode=0755,soft,\
nounix,serverino,mapposix,rsize=4194304,wsize=4194304,bsize=1048576,\
echo_interval=60,actimeo=1,user=ktom)
```

Listing 15.57 Die gemounteten Freigaben über »`pam_mount`«

Mit der Konfiguration von `pam_mount` ist es jetzt möglich, Freigaben des Samba-Servers auf den Linux-Clients zu verwenden.

15.8.3 Umstellen des grafischen Logins

Da es sich bei der Installation des Clients um ein openSUSE Leap mit grafischer Umgebung handelt, soll jetzt der Login-Manager noch so umgestellt werden, dass eine Anmeldung der Domänenbenutzer am Client möglich ist. Denn die Standardeinstellung, die im Übrigen bei allen hier im Buch eingesetzten Distributionen identisch ist, ist immer so, dass sich nur lokale Benutzer anmelden können.

Damit sich die Benutzer am Active Directory anmelden können, passen Sie den `lightdm` über die Datei `/etc/sysconfig/displaymanager` an. Dort setzen Sie die Variable `DISPLAYMANAGER_AD_INTEGRATION="yes"`. Nach einem Neustart des Systems funktioniert die Anmeldung am System.

Auf einem Ubuntu-Client passen Sie die Datei `/etc/lightdm/lightdm.conf` an. Setzen Sie hier die Variable `greeter-show-manual-login=true`. Nach dem nächsten Neustart des Systems können Sie sich dann mit dem Domänenbenutzer anmelden.

Auf dem Linux-Mint-System, das wir hier zum Testen verwenden, kann die Einstellung für den Anmeldebildschirm auf der Benutzeroberfläche durchgeführt werden. Aktivieren Sie hierzu unter EINSTELLUNGEN•ANMELDEFENSTER die Option MANUELLES ANMELDEN ERLAUBEN im Karteireiter BENUTZER. Abbildung 15.19 zeigt die Einstellung.

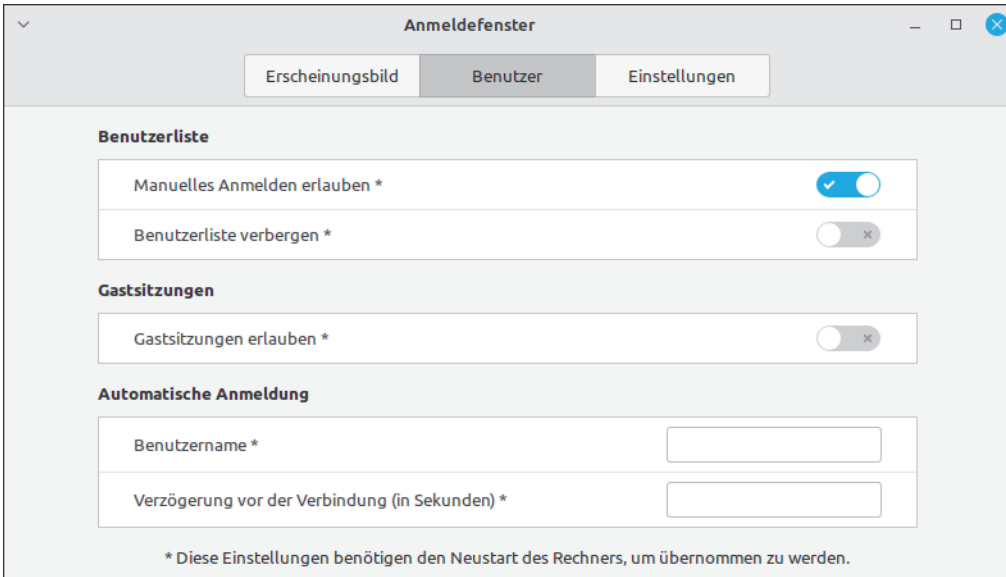


Abbildung 15.19 Aktivierung der Benutzeranmeldung

Denken Sie bei allen Linux-Clients daran, dass der Benutzer, der sich anmeldet, immer auch ein Heimatverzeichnis benötigt, um die Desktopumgebung speichern zu können.

Damit ist die Integration eines Linux-Clients in die Domäne abgeschlossen. Welchen Client Sie verwenden, spielt dabei keine Rolle, die Schritte sind immer identisch.

15.9 Zusätzliche Server in der Domäne

Ein Server in einer Domäne reicht meist nicht aus: Sie wollen und sollten Ihre Daten auf jeden Fall auf einem zusätzlichen Fileserver ablegen und nicht auf einem Domaincontroller – vor allem dann, wenn Sie neben den Windows-Clients auch noch Linux-Clients im Netz einsetzen. Denn der Domaincontroller hat immer ein eigenes ID-Mapping, sodass es bei der Vergabe der Rechte zu Problemen kommen kann.

Auch möchten Sie sicher die Anmeldung der Benutzer durch den Einsatz eines zweiten Domaincontrollers ausfallsicher gestalten.

15.9.1 Einen Fileserver einrichten

Wollen Sie zu Ihrem Domaincontroller noch einen Fileserver in Ihrer Domäne installieren, binden Sie diesen erst als Client in die Domäne ein. Erst wenn der zukünftige Fileserver ein Client in der Domäne ist, können Sie mit der Konfiguration als Fileserver beginnen. Ob Sie hier die Samba-Pakete der Distribution nutzen, ist Ihnen überlassen. Für den Fileserver benötigen Sie nicht unbedingt die aktuellste Samba-Version. Um den Linux-Client zu einem Samba-Client zu machen, müssen Sie als Erstes die Datei `/etc/samba/smb.conf` so wie in Listing 15.58 anlegen:

```
[global]
    workgroup = example
    realm = EXAMPLE.NET
    security = ADS
    winbind refresh tickets = yes
    winbind use default domain = yes
    template shell = /bin/bash
    idmap config * : range = 100000 - 199999
    idmap config EXAMPLE : backend = rid
    idmap config EXAMPLE : range = 1000000 - 1999999
    inherit acls = yes
    vfs objects = acl_xattr
```

Listing 15.58 Die »smb.conf« auf dem Client

Die zwei zusätzlichen Parameter `inherit acls = yes` und `vfs objects = acl_xattr` sorgen dafür, dass Sie die Berechtigungen auf dem Samba-Fileserver unter Windows verwalten können. Nur mit diesen Parametern können Sie die Rechte Windows-konform verwalten. Wenn Sie bis zu diesem Punkt die Heimatverzeichnisse und die Profilverzeichnisse der Benutzer auf dem Domaincontroller abgelegt haben, wird es jetzt Zeit, diese auf den Fileserver zu verschieben. Selbstverständlich können Sie die Freigaben wie gewohnt in der `smb.conf` einrichten. Besser ist es aber, die Freigaben in Zukunft in der Registry abzulegen. Das hat verschiedene Vorteile:

- ▶ Änderungen an der `smb.conf` werden immer an alle verbundenen Clients übertragen. Das heißt, wenn Sie in Ihrem Netz mehrere Hundert Clients betreiben, wird bei jeder Änderung die gesamte Datei an alle Clients übertragen. Je größer die Datei ist und je mehr Änderungen Sie durchführen, umso mehr Traffic erzeugen Sie in Ihrem Netzwerk. Wenn Sie die Registry verändern, wird nur die jeweilige Änderung an die Clients übertragen.
- ▶ Mehrere Admins können die Konfiguration gleichzeitig bearbeiten.
- ▶ Die Registry kann auch von einem Windows-Client bearbeitet werden. Eine `ssh`-Verbindung ist dann nicht nötig. Die Bearbeitung der Registry können Sie dort mit dem Programm `regedit` durchführen.

Konfiguration in der Registry

Nachdem Sie den Fileserver erfolgreich in die Domäne aufgenommen haben und alle Benutzer auf dem System zur Verfügung stehen, wollen wir vor dem Einrichten der ersten Freigabe die Konfiguration in die Registry verlagern. Im ersten Schritt kopieren Sie die derzeitige Konfiguration aus der *smb.conf* in die Registry (siehe Listing 15.59):

```
root@fs-01:~# net conf import /etc/samba/smb.conf
root@fs-01:~# net conf list
[global]
    realm = EXAMPLE.NET
    registry shares = Yes
    security = ADS
    template shell = /bin/bash
    winbind refresh tickets = Yes
    winbind use default domain = Yes
    workgroup = EXAMPLE
    idmap config example : range = 1000000 - 1999999
    idmap config example : backend = rid
    idmap config * : range = 100000 - 199999
    idmap config * : backend = tdb
    inherit acls = Yes
    vfs objects = acl_xattr
```

Listing 15.59 Umstellen der Konfiguration auf die Registry

Mit dem ersten Kommando `net conf import <Datei>` kopieren Sie eine bestehende Konfiguration in die Registry. Anschließend lassen Sie sich den Inhalt der Registry mit dem Kommando `net conf list` anzeigen. Dieses Kommando zeigt Ihnen die Registry im Format der *smb.conf* an. Durch den Einsatz eines Redirectors können Sie die Ausgabe auch in eine Datei umleiten. So haben Sie die Möglichkeit, die Konfiguration wieder auf die *smb.conf* umzustellen oder die Konfiguration anzupassen. Denn wenn Sie die Datei editieren, können Sie die Änderung wieder mit `net conf import <Datei>` in die Registry einspielen.



Bei der Verwendung von `net conf import` wird immer die gesamte Registry überschrieben und nicht nur die Änderungen.

Im nächsten Schritt konfigurieren Sie den Samba so, dass er in Zukunft die Konfiguration aus der Registry liest. Samba liest grundsätzlich immer als Erstes die Datei *smb.conf* ein. Daher benötigen Sie auch weiterhin eine *smb.conf*, nur werden Sie dort nur noch auf die Registry verweisen. Erstellen Sie jetzt eine neue Datei */etc/samba/smb.conf* mit diesem Inhalt:

```
[global]
    config backend = registry
```

Listing 15.60 Die neue »smb.conf«

Im Anschluss können Sie die Umstellung mit dem Kommando `testparm` überprüfen. Sie brauchen keine der Dienste für Samba neu zu starten (siehe Listing 15.61).

```
root@fs-01:/etc/samba# testparm
Load smb config files from /etc/samba/smb.conf
lp_load_ex: changing to config backend registry
Loaded services file OK.
Weak crypto is allowed by GnuTLS (e.g. NTLM as a compatibility fallback)
```

```
Server role: ROLE_DOMAIN_MEMBER
```

Press enter to see a dump of your service definitions

```
# Global parameters
[global]
    realm = EXAMPLE.NET
    registry shares = Yes
    security = ADS
    template shell = /bin/bash
    winbind refresh tickets = Yes
    winbind use default domain = Yes
    workgroup = EXAMPLE
    idmap config example : range = 1000000 - 1999999
    idmap config example : backend = rid
    idmap config * : range = 100000 - 199999
    idmap config * : backend = tdb
    inherit acls = Yes
    vfs objects = acl_xattr
```

Listing 15.61 Ausgabe von »testparm«

Die Zeile `lp_load_ex: changing to config backend registry` zeigt, dass die Konfiguration jetzt aus der Registry geladen wird. Sie haben auch die Möglichkeit, sich die einzelnen Schlüssel der Registry anzeigen zu lassen. Dazu nutzen Sie das Kommando `net registry` so wie in Listing 15.62:

```
root@fs-01:~# net registry enumerate 'hklm\software\samba\smbconf\global'
Valuename = workgroup
Type      = REG_SZ
Value     = "example"

Valuename = realm
Type      = REG_SZ
Value     = "EXAMPLE.NET"
```

```
Valuename = security
Type      = REG_SZ
Value     = "ADS"

Valuename = winbind refresh tickets
Type      = REG_SZ
Value     = "yes"
Valuename = winbind use default domain
Type      = REG_SZ
Value     = "yes"

Valuename = template shell
Type      = REG_SZ
Value     = "/bin/bash"

Valuename = idmap config * : range
Type      = REG_SZ
Value     = "100000 - 199999"

Valuename = idmap config EXAMPLE : backend
Type      = REG_SZ
Value     = "rid"

Valuename = idmap config EXAMPLE : range
Type      = REG_SZ
Value     = "1000000 - 1999999"

Valuename = inherit acls
Type      = REG_SZ
Value     = "yes"

Valuename = vfs objects
Type      = REG_SZ
Value     = "acl_xattr"
```

Listing 15.62 Ausgabe der Registry mit »net registry enumerate«

Mit dem Kommando net können Sie auch Einträge in der Registry einzeln eintragen, löschen oder verändern. Im nächsten Abschnitt wollen wir Ihnen zeigen, wie Sie Freigaben in der Registry anlegen können.

Anlegen von Freigaben auf dem Server

Jetzt sollen die Freigaben auf dem Fileserver eingerichtet werden. In unserem Beispiel sollen die beiden Freigaben *users* und *profile* angelegt werden (siehe Listing 15.63).

```

root@fs-01:~# net conf addshare users /home/EXAMPLE writeable=y \
  guest_ok=n "Home-Dirs"

root@fs-01:~# net conf setparm users "browsable" "no"
root@fs-01:~# net conf setparm users "create mask" "700"
root@fs-01:~# mkdir /profile
root@fs-01:~# chmod 1770 /profile/
root@fs-01:~# chgrp 'domain users' /profile/
root@fs-01:~# net conf addshare profile /profile writeable=y \
  guest_ok=n "User Profile"
root@fs-01:~# net conf setparm profile "browsable" "no"

```

Listing 15.63 Anlegen der Freigaben für die Heimatverzeichnisse und Profile

Mit dem Kommando `net conf list` können Sie die gesamte Konfiguration aus der Registry auslesen lassen. Sie bekommen dabei die vollständige Konfiguration im Stil der `smb.conf` angezeigt. Nachdem Sie die Freigaben erstellt und die entsprechenden Rechte im Dateisystem gesetzt haben, können Sie die Benutzer in den RSAT so anpassen, dass diese in Zukunft ihre Daten vom Fileserver beziehen. Wollen Sie die Schlüssel der Registry einzeln bearbeiten, haben Sie drei Möglichkeiten:

- ▶ über die Kommandozeile mit den Kommandos `net conf` und `net registry` – Mit der Option `--help` bekommen Sie eine Hilfe zu den Subkommandos.
- ▶ mit dem Tool `samba-regedit` – Dieses Tool starten Sie auf dem Server. Sie bekommen dann eine auf `ncurses` basierende grafische Oberfläche zur Bearbeitung der Registry.
- ▶ über Windows mithilfe des Programms *Regedit*.

Anlegen von Freigaben über Windows

Die beiden Freigaben aus dem vorherigen Beispiel sind benutzerbezogen. Das heißt, jeder Benutzer, der auf die Freigabe zugreift, hat einen eigenen Bereich, auf den nur er Zugriff hat. Wenn Sie aber Freigaben einrichten wollen, auf die verschiedene Gruppen unterschiedliche Zugriffsrechte erhalten sollen, dann ist der nun folgende Weg der bessere.

Im globalen Bereich der `smb.conf` wurden die beiden Parameter `inherit acls = yes` und `vfs objects = acl_xattr` gesetzt. Durch diese beiden Parameter wird dafür gesorgt, dass die Berechtigungen in den Freigaben den Berechtigungen eines Windows-Systems entsprechen. Dies gilt aber nur dann, wenn vom Anlegen der Freigaben, die später von den Anwendern genutzt werden, bis hin zur Vergabe der Berechtigungen alle Schritte unter Windows durchgeführt werden.

Um die Freigaben und Berechtigungen in Zukunft so verwalten zu können, benötigen Sie als Erstes einen Einstiegspunkt in Ihr Dateisystem, und zwar in Form einer administrativen Freigabe. Legen Sie hierzu ein Verzeichnis so wie in Listing 15.64 an:

```
root@fs-01:~# mkdir /admin-share
root@fs-01:~# chgrp 'domain admins' /admin-share/
root@fs-01:~# chmod 775 /admin-share/
```

Listing 15.64 Anlegen des Verzeichnisses für die administrative Freigabe

Erzeugen Sie dazu jetzt die administrative Freigabe, so wie Sie sie in Listing 15.65 sehen:

```
root@fs-01:~# net conf addshare admin-share /admin-share writeable=y \
  guest_ok=n "Admin only"
root@fs-01:~# net conf setparm admin-share "browsable" "no"
root@fs-01:~# net conf setparm admin-share "read only" "no"
root@fs-01:~# net conf setparm admin-share "administrative share" "yes"
```

Listing 15.65 Anlegen der administrativen Freigabe

Durch den Parameter `administrative share = yes` erzeugen Sie eine Freigabe mit denselben Eigenschaften, wie sie die *LW\$*-Freigaben unter Windows haben.

Wenn Sie sich jetzt auf einem Windows-Client der Domäne als ein Mitglied der Gruppe *domain admins* anmelden und sich mit der Freigabe verbinden, können Sie die Verzeichnisse anlegen, die Sie anschließend für Ihre Benutzer freigeben wollen. Sie können bei den so angelegten Verzeichnissen anschließend die Rechte genau so vergeben, wie Sie es sonst auf einem Windows-Server machen. Alle Regeln eines Windows-Systems werden dann bei den Berechtigungen übernommen – selbst die Tatsache, dass der Administrator nicht mehr in ein Verzeichnis wechseln kann, wenn er keinen Eintrag in den ACLs des Verzeichnisses besitzt. Der große Vorteil dieses Vorgehens ist: Wenn Sie alle Zugriffe von allen Client-Betriebssystemen jetzt ausschließlich über SMB erlauben, bleiben die Rechte immer erhalten, egal von welchem Betriebssystem aus ein Benutzer zugreift. Selbst wenn der Benutzer mit unterschiedlichen Betriebssystemen zugreift, bleiben die Berechtigungen erhalten.

15.9.2 Ein zusätzlicher Domaincontroller

Um einen zusätzlichen Domaincontroller in Ihrem Netz einrichten zu können, installieren Sie als Erstes einen Linux-Rechner in der Grundinstallation und anschließend wieder alle benötigten Pakete, die Sie auch schon beim ersten Domaincontroller installiert haben. Verwenden Sie für den zweiten Domaincontroller immer dieselbe Distribution wie für den ersten Domaincontroller, um später immer dieselbe Samba-Version auf allen Domaincontrollern zu haben. Sorgen Sie immer dafür, dass alle Domaincontroller dieselbe Version nutzen, das macht die Verwaltung der Systeme einfacher.

Für den neuen Domaincontroller ist es nicht mehr notwendig, die DNS-Einträge im Voraus anzulegen. Beim Join werden die benötigten Einträge automatisch erzeugt. Aber um Ihnen zu zeigen, wie Sie DNS-Einträge auch manuell erstellen können, werden wir die Einträge hier vorab von Hand anlegen. Auch werden wir gleichzeitig eine Reverse-Zone im DNS erzeugen.

Sie können alle Einträge auch über den *DNS-Manager* der RSAT erstellen. Wir wollen Ihnen hier aber zeigen, wie Sie die Einträge auch auf der Kommandozeile anlegen können. In Listing 15.66 sehen Sie die passenden Kommandos, um die entsprechenden Einträge im DNS anzulegen. Alle Kommandos führen Sie auf dem bereits bestehenden Domaincontroller aus:

```
root@addc-01:~# kinit administrator
administrator@EXAMPLE.NET's Password:

root@addc-01:~# samba-tool dns add addc-01 example.net addc-02 A \
                    192.168.56.82 -N
Record added successfully

root@addc-01:~# samba-tool dns add addc-01 56.168.192.in-addr.arpa 82 \
                    PTR addc-02.example.net
Record added successfully
```

Listing 15.66 Den DNS-Server einrichten

Sorgen Sie dafür, dass der zweite DNS-Server vor dem Beitritt zur Domäne den ersten Domaincontroller als DNS-Server verwendet. Nachdem Sie den zweiten Domaincontroller komplett konfiguriert haben, sollten Sie die Namensauflösung aber auf jeden Fall wieder so einstellen, dass jeder Domaincontroller die Namensauflösung immer selbst realisiert.



Bei Revers-Zonen zu beachten

Wenn Sie *Reverse-Zonen* anlegen, beachten Sie bitte, dass als *NS-record* immer nur der Domaincontroller eingetragen wird, auf dem die Zone erstellt wurde.

Im Gegensatz zu den *Forward-Zonen* wird kein weiterer *NS-Record* erzeugt, wenn ein zusätzlicher Domaincontroller in die Domäne aufgenommen wird. Auch beim Entfernen dieser Domaincontroller wird kein neuer NS-Record erzeugt. Das führt dazu, dass die Zone keinen NS-Record mehr besitzt und der *Bind9* wird nicht mehr starten. Sollte es Ihnen trotzdem passieren, dass Sie eine Revers-zone ohne einen NS-Record haben, können Sie den fehlenden Eintrag wie folgt wiederherstellen:

- ▶ Wechseln Sie mit dem Kommando `samba_upgradedns` zurück auf den internen DNS-Server.
- ▶ Fügen Sie den Parameter `dns` in der `smb.conf` zum Parameter `server services =` hinzu.
- ▶ Starten Sie den Domaincontrollerdienst neu.
- ▶ Erstellen Sie einen neuen NS-Record mit dem Kommando `samba-tool dns add addc-01 56.168.192.in-addr.arpa @ NS addc-02.example.net -U administrator`.
- ▶ Führen Sie das Kommando `samba_upgradedns --dns-backend=BIND9_DLZ` aus.
- ▶ Entfernen Sie den Wert `dns` wieder aus der Zeile `server services =` der `smb.conf`.
- ▶ Starten Sie den `bind9` und den Domaincontrollerdienst neu.

Jetzt hat Ihre *Reverse-Zone* wieder einen *NS-Record*. Besitzen Sie mehrere *Reverse-Zonen*, prüfen Sie auch hier die *NS-Records* und passen Sie diese eventuell an.

15.9.3 Konfiguration des zweiten DC

Kopieren Sie als Erstes die Datei */etc/krb5.conf* vom ersten Domaincontroller auf den neuen Domaincontroller, da sonst die Authentifizierung über Kerberos nicht möglich ist und der Beitritt zur Domäne fehlschlägt. Mit dem Kommando *samba-tool* fügen Sie jetzt den neuen Domaincontroller, so wie in Listing 15.67, zur Domäne hinzu. Kopieren Sie anschließend die Datei */var/lib/samba/private/krb5.conf* ins Verzeichnis */etc*.

```
root@addc-02:~# samba-tool domain join --dns-backend=BIND9_DLZ example.net DC \
--realm=example.net -Uadministrator
INFO 2022-12-27 19:53:16,860 pid:3946 /usr/lib/python3/dist-packages/\
samba/join.py #105: Finding a writeable DC for domain 'example.net'
INFO 2022-12-27 19:53:16,874 pid:3946 /usr/lib/python3/dist-packages/\
samba/join.py #107: Found DC addc-01.example.net
Password for [WORKGROUP\administrator]:
[...]
: Sending DsReplicaUpdateRefs for all the replicated partitions
: Setting isSynchronized and dsServiceName
: Setting up secrets database
: See /var/lib/samba/bind-dns/named.conf for an example configuration \
include file for BIND
: and /var/lib/samba/bind-dns/named.txt for further documentation \
required for secure DNS updates
: Joined domain EXAMPLE (SID S-1-5-21-2492729547-2181408212-3299912776)\
as a DC
```

Listing 15.67 Beitritt zur Domäne des zweiten Domaincontrollers

15.9.4 Einrichten des Nameservers

Jetzt können Sie den Bind9 auch auf dem zweiten Domaincontroller konfigurieren. Für die Konfiguration finden Sie im Verzeichnis */etc/bind* die beiden Dateien *named.conf.options* und *named.conf.local*. Am einfachsten ist es, Sie kopieren die beiden Dateien vom ersten Domaincontroller, denn die beiden Dateien sind auf beiden Domaincontrollern identisch.

Unter Ubuntu wird der Bind9 über *AppArmor* in eine Art chroot-Umgebung gezwungen. Das heißt, der Bind9 kann nur auf die Dateien zugreifen, die von *AppArmor* freigegeben wurden. Damit der Bind9 überhaupt Zugriff auf die Datenbanken des Active Directory bekommen kann, passen Sie jetzt die Konfiguration von *AppArmor* in der Datei */etc/AppArmor.d/usr.sbin.named* so wie in Listing 15.68 an:

```

/etc/bind/** r,
/var/lib/bind/** rw,
/var/lib/bind/ rw,
/var/cache/bind/** lrw,
/var/lib/samba/** rwmk,
/usr/lib/x86_64-linux-gnu/** rwmk,
/dev/urandom rwmk,

```

Listing 15.68 Einstellungen von AppArmor

Aufgrund der drittletzten Zeile gewährt AppArmor dem Bind9 Zugriff auf das Verzeichnis */var/lib/samba/private/* und alle Unterverzeichnisse. Nachdem Sie die Datei angepasst haben, starten Sie den Dienst mit dem Kommando `systemctl restart apparmor.service` neu.

Bei einigen Distributionen sind diese Einträge mittlerweile vorhanden. Um sicherzugehen, prüfen Sie die entsprechenden Datei.



Jetzt können Sie den Bind9 neu starten und anschließend in der Log-Datei nach den Zeilen schauen, die den erfolgreichen Zugriff auf das Active Directory melden. In Listing 15.69 sehen Sie die wichtigen Zeilen:

```

root@addc-02:~# systemctl restart bind9
[...]
root@addc-02:~# tail -n 200 /var/log/syslog
[...]
: Loading 'AD DNS Zone' using driver dlopen
: samba_dlz: started for DN DC=example,DC=net
: samba_dlz: starting configure
: samba_dlz: configured writeable zone 'example.net'
: samba_dlz: configured writeable zone '56.168.192.in-addr.arpa'
: samba_dlz: configured writeable zone '_msdcs.example.net'

```

Listing 15.69 Erster Start des Bind9

Erst wenn Sie im Log-File diese Einträge finden, ist der Bind9 vollständig konfiguriert und hat alle Rechte, um auf das Active Directory zugreifen zu können.

Wie schon beim ersten Domaincontroller ist es hier notwendig, dass Sie den Samba-Domaincontroller-Dienst erst aktivieren und die Dienste `smbd`, `nmbd` und `winbind` deaktivieren. Ohne diese Schritte würde Samba nicht als Domaincontroller starten. In Listing 15.70 sehen Sie die erforderlichen Kommandos:

```

root@addc-02:~# systemctl stop smbd nmbd winbind
root@addc-02:~# systemctl disable smbd nmbd winbind
Synchronizing state of smbd.service with SysV service script with \
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable smbd

```

```
Synchronizing state of nmbd.service with SysV service script with \
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable nmbd
Synchronizing state of winbind.service with SysV service script with \
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable winbind
```

```
root@addc-02:~# systemctl unmask samba-ad-dc
Removed /etc/systemd/system/samba-ad-dc.service.
```

```
root@addc-02:~# systemctl start samba-ad-dc
```

```
root@addc-02:~# systemctl enable samba-ad-dc
Synchronizing state of samba-ad-dc.service with SysV service script with \
/lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable samba-ad-dc
```

Listing 15.70 Aktivieren der Domaincontroller-Funktion

Anschließend können Sie mit dem Kommando `ps ax | grep samba` prüfen, ob der Samba-Domaincontroller-Dienst gestartet wurde. In Listing 15.71 sehen Sie das Ergebnis:

```
root@addc-02:~# ps ax | grep samba
4784 ?      Ss      0:00 samba: root process
4785 ?      S       0:00 samba: tfork waiter process(4786)
4786 ?      S       0:00 samba: task[s3fs] pre-fork master
4787 ?      S       0:00 samba: tfork waiter process(4788)
4788 ?      S       0:00 samba: task[rpc] pre-fork master
4789 ?      S       0:00 samba: tfork waiter process(4791)
4790 ?      S       0:00 samba: tfork waiter process(4793)
4791 ?      S       0:00 samba: task[nbt] pre-fork master
4792 ?      S       0:00 samba: tfork waiter process(4794)
4794 ?      S       0:00 samba: task[wrepl] pre-fork master
4795 ?      S       0:00 samba: tfork waiter process(4797)
4796 ?      S       0:00 samba: tfork waiter process(4799)
4797 ?      S       0:01 samba: task[ldap] pre-fork master
4798 ?      S       0:00 samba: tfork waiter process(4801)
4799 ?      S       0:00 samba: task[rpc] pre-forked worker(0)
4800 ?      S       0:00 samba: tfork waiter process(4803)
4801 ?      S       0:00 samba: task[cldap] pre-fork master
4802 ?      S       0:00 samba: tfork waiter process(4805)
4803 ?      S       0:00 samba: task[rpc] pre-forked worker(1)
4804 ?      S       0:00 samba: tfork waiter process(4807)
4805 ?      S       0:00 samba: task[kdc] pre-fork master
```

```

4806 ?      S      0:00 samba: tfork waiter process(4810)
4807 ?      S      0:00 samba: task[rpc] pre-forked worker(2)
4808 ?      S      0:00 samba: tfork waiter process(4812)
4809 ?      S      0:00 samba: tfork waiter process(4813)
4810 ?      S      0:00 samba: task[drepl] pre-fork master
4811 ?      S      0:00 samba: tfork waiter process(4814)
4812 ?      S      0:00 samba: task[rpc] pre-forked worker(3)
4813 ?      S      0:00 samba: task[kdc] pre-forked worker(0)
4814 ?      S      0:00 samba: task[winbindd] pre-fork master
4815 ?      S      0:00 samba: tfork waiter process(4816)
4816 ?      S      0:00 samba: task[ntp_signd] pre-fork master
4817 ?      S      0:00 samba: tfork waiter process(4820)
4818 ?      S      0:00 samba: tfork waiter process(4822)
4819 ?      S      0:00 samba: tfork waiter process(4824)
4820 ?      S      0:00 samba: task[kcc] pre-fork master
4821 ?      S      0:00 samba: tfork waiter process(4825)
4822 ?      S      0:00 samba: task[kdc] pre-forked worker(1)
4823 ?      S      0:00 samba: tfork waiter process(4827)
4825 ?      S      0:00 samba: task[dnupdate] pre-fork master
4827 ?      S      0:00 samba: task[kdc] pre-forked worker(2)
4828 ?      S      0:00 samba: tfork waiter process(4829)
4829 ?      S      0:00 samba: task[kdc] pre-forked worker(3)
4886 ?      S      0:00 samba: tfork waiter process(4887)
4887 ?      S      0:00 samba: task[ldap] pre-forked worker(0)
4888 ?      S      0:00 samba: tfork waiter process(4889)
4889 ?      S      0:00 samba: task[ldap] pre-forked worker(1)
4890 ?      S      0:00 samba: tfork waiter process(4892)
4892 ?      S      0:00 samba: task[ldap] pre-forked worker(2)
4893 ?      S      0:00 samba: tfork waiter process(4894)
4894 ?      S      0:00 samba: task[ldap] pre-forked worker(3)

```

Listing 15.71 Erster Start des Domaincontrollers

Sorgen Sie jetzt auf jeden Fall dafür, dass der neue Domaincontroller seine eigene IP-Adresse als Resolver eingetragen hat. Bevor Sie die Funktionstests durchführen, starten Sie die Server einmal komplett neu, um zu testen, ob alle Dienste auch bei einem Systemstart sauber gestartet werden.

15.9.5 Testen der Replikation

Nachdem der Server wieder erreichbar ist, folgen jetzt verschiedene Tests, mit denen Sie prüfen können, ob die beiden Domaincontroller ihre Informationen und Änderungen auch replizieren. Starten wollen wir mit dem Konsistenztest, den Sie in Listing 15.72 sehen:

```
root@addc-02:~# kinit administrator
administrator@EXAMPLE.NET's Password
```

```
root@addc-02:~# samba-tool drs kcc addc-02.example.net
Consistency check on addc-02.example.net successful.
root@addc-02:~# samba-tool drs kcc addc-01.example.net
Consistency check on addc-01.example.net successful.
```

Listing 15.72 Durchführung des Konsistenztests

Mit diesem Test stellen Sie sicher, dass die Datenbanken auf beiden Domaincontrollern konsistent sind. Mit dem nächsten Test (siehe Listing 15.73) können Sie die Replikation auf beiden Domaincontrollern prüfen:

```
root@addc-02:~# samba-tool drs showrepl --summary
[ALL GOOD]
```

Listing 15.73 Testen der Replikation

Hier wird durch den Parameter `--summary` nur das Gesamtergebnis angezeigt. Gerade wenn Sie mehr als zwei Domaincontroller verwalten, sorgt das für eine Übersichtlichkeit der Ergebnisse. Sollte eine Replikation nicht ordnungsgemäß durchführbar sein, würden Sie auch nur diese Replikation als Fehler angezeigt bekommen.

Jetzt testen Sie noch, ob alle Services auch von beiden Domaincontrollern über DNS angeboten werden. Dazu fragen Sie die verschiedenen *Service-Records* auf beiden Domaincontrollern ab. In Listing 15.74 sehen Sie alle Tests auf beiden Domaincontrollern und die Ergebnisse:

```
root@addc-02:~# host -t srv _ldap._tcp.example.net
_ldap._tcp.example.net has SRV record 0 100 389 addc-01.example.net.
_ldap._tcp.example.net has SRV record 0 100 389 addc-02.example.net.
```

```
root@addc-02:~# host -t srv _kerberos._tcp.example.net
_kerberos._tcp.example.net has SRV record 0 100 88 addc-02.example.net.
_kerberos._tcp.example.net has SRV record 0 100 88 addc-01.example.net.
```

```
root@addc-02:~# host -t srv _gc._tcp.example.net
_gc._tcp.example.net has SRV record 0 100 3268 addc-02.example.net.
_gc._tcp.example.net has SRV record 0 100 3268 addc-01.example.net.
```

```
root@addc-01:~# host -t srv _ldap._tcp.example.net
_ldap._tcp.example.net has SRV record 0 100 389 addc-02.example.net.
_ldap._tcp.example.net has SRV record 0 100 389 addc-01.example.net.
root@addc-01:~# host -t srv _kerberos._tcp.example.net
_kerberos._tcp.example.net has SRV record 0 100 88 addc-02.example.net.
_kerberos._tcp.example.net has SRV record 0 100 88 addc-01.example.net.
```

```
root@addc-01:~# host -t srv _gc._tcp.example.net
_gc._tcp.example.net has SRV record 0 100 3268 addc-01.example.net.
_gc._tcp.example.net has SRV record 0 100 3268 addc-02.example.net.
```

Listing 15.74 Testen der Service-Records

Führen Sie die Tests unbedingt auf beiden Domaincontrollern durch! Nur so können Sie sicher sein, dass alle Domaincontroller alle Service-Records kennen. Sollten Sie auf einem der Server nur einen der beiden Domaincontroller sehen, prüfen Sie Ihre DNS-Umgebung. Testen Sie, ob ein `samba_dnsupdate --verbose --all-names` keinen Fehler ausgibt. Sollte es zu Fehlern auf einem der Domaincontroller kommen, gibt es Probleme mit Ihrem DNS-Setup.

15.9.6 Weitere Tests

Zusätzlich zu den vorher beschriebenen Tests können Sie noch weitere Tests durchführen, um die Replikation zu prüfen. Sie können die Replikation aber auch manuell anstoßen, um dann das Ergebnis zu prüfen. Ein sehr einfacher und praktischer Test besteht darin, dass Sie auf der Kommandozeile eines Domaincontrollers einen neuen Benutzer oder eine neue Gruppe anlegen und dann auf dem zweiten Domaincontroller prüfen, ob der Benutzer oder die Gruppe auch dort sichtbar ist. Wenn das der Fall ist, können Sie mit Sicherheit davon ausgehen, dass die Replikation funktioniert. Wollen Sie die Replikation von Hand ausführen, um eventuell existierende Fehler zu finden, können Sie die Replikation mit `samba-tool` von Hand starten. In Listing 15.75 sehen Sie diesen Vorgang:

```
root@adminbuch-2:~# samba-tool drs replicate
Usage: samba-tool drs replicate <destinationDC> <sourceDC> <NC> [options]

root@adminbuch-2:~# samba-tool drs replicate addc-01 addc-02 dc=example,dc=net
Replicate from addc-02 to addc-01 was successful.

root@adminbuch-2:~# samba-tool drs replicate addc-02 addc-01 dc=example,dc=net
Replicate from addc-01 to addc-02 was successful.
```

Listing 15.75 Manueller Start der Replikation

Die Replikation wurde hier in beide Richtungen erfolgreich getestet. Anders als bei anderen Linux-Kommandos, bei denen immer zuerst die Quelle und dann das Ziel angegeben wird, müssen Sie hier immer erst das Ziel und dann die Quelle angeben.

15.9.7 Einrichten des Zeitserver

Auch auf dem zweiten Domaincontroller richten Sie wieder einen Zeitserver ein. Da sich ein Client an jedem beliebigen Domaincontroller authentifizieren kann, müssen immer

alle Domaincontroller in der Lage sein, die Zeitinformation signiert an alle Clients geben zu können. Verfahren Sie hier genau so wie auf dem ersten Domaincontroller. Installieren Sie das Paket *ntp*, und kopieren Sie dann die Datei */etc/ntp.conf* vom ersten Domaincontroller auf den zweiten Domaincontroller. Anschließend sorgen Sie noch dafür, dass die Berechtigungen am Verzeichnis */var/lib/samba//ntp_signd* stimmen (tragen Sie die Gruppe *ntp* als besitzende Gruppe ein). Starten Sie dann den Zeitserver neu, und die Installation des zweiten Domaincontrollers ist so weit abgeschlossen. Was jetzt noch fehlt, ist die Replikation der Freigabe *sysvol*. Darum soll es im nächsten Abschnitt gehen.

15.10 Die Replikation der Freigabe »sysvol« einrichten

Zurzeit gibt es noch keine Möglichkeit der *File System Replication*, deshalb wird die Freigabe *sysvol* nicht mit repliziert. Die Freigaben *sysvol* und *netlogon* müssen aber auf jedem Domaincontroller existieren, denn sonst könnten die Gruppenrichtlinien nicht bei jeder Anmeldung abgearbeitet werden. Solange die Replikation des Dateisystems noch nicht von Samba durchgeführt werden kann, müssen Sie dafür sorgen, dass die Replikation der Freigaben in regelmäßigen Abständen durchgeführt wird. Am einfachsten lässt sich das über *rsync* realisieren. Im nächsten Abschnitt zeigen wir Ihnen, wie Sie *rsync* einrichten können und worauf Sie dabei zu achten haben.

15.10.1 Einrichten des *rsync*-Servers

Prüfen Sie im ersten Schritt, welcher Domaincontroller die *Flexible Single Master Operations*-(FSMO)-Rolle *PdcEmulationMasterRole* hält, denn nur auf diesem Server dürfen Sie in Zukunft Änderungen an den Gruppenrichtlinien und den Logon-Skripten durchführen. Von dort werden dann die anderen Domaincontroller ihre benötigten Daten aus der Freigabe *sysvol* holen. Den entsprechenden Server können Sie so wie in Listing 15.76 herausfinden:

```
root@addc-02:/etc# samba-tool fsmo show
SchemaMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,CN=Default-\  
  First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
InfrastructureMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,\  
  CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
RidAllocationMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,\  
  CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
PdcEmulationMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,\  
  CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
DomainNamingMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,\  
  CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
DomainDnsZonesMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,\  
  CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
```



```
ForestDnsZonesMasterRole owner: CN=NTDS Settings,CN=ADDC-01,CN=Servers,\
  CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=example,DC=net
```

Listing 15.76 Ermittlung des PDC-Masters

Wie Sie hier sehen, ist der erste Domaincontroller derjenige, der die FSMO-Rolle hält. Auf ihm wird jetzt der *rsync-Server* eingerichtet.

15.10.2 Einrichten von *rsync* auf dem PDC-Master

Sie haben zwei verschiedene Möglichkeiten, *rsync* als Server zu starten. Standardmäßig wird der *rsync* über den *systemd* gestartet, Sie benötigen lediglich eine Konfigurationsdatei für den Server und können dann den Dienst mit `systemctl restart rsync` neu starten. Die andere Möglichkeit ist die, dass Sie den *rsync*-Server über den *xinetd* starten. Das hat den Vorteil, dass Sie in der Konfiguration des *xinetd* die IP-Adressen festlegen können, die sich mit dem Dienst verbinden dürfen. Die Konfiguration für den *rsync*-Server ist in beiden Fällen identisch. Für die Einrichtung über den *systemd* benötigen Sie keine zusätzlichen Schritte.

Einrichtung über *xinetd*

Auf dem PDC-Master brauchen Sie neben dem Programm *rsync* auch noch den *xinetd*, um den *rsync-Server* starten zu können. Installieren Sie unter Debian und Ubuntu die Pakete *rsync* und *xinetd* auf dem entsprechenden DC. Deaktivieren Sie den Start des *rsync*-Servers über den *Systemd* mit dem Kommando `systemctl disable --now rsync.service`. Dieses Kommando verhindert den Neustart beim Booten des Systems und stoppt den Dienst sofort. Anschließend erstellen Sie unter Debian und Ubuntu für den *xinetd* eine Konfigurationsdatei *rsync* im Verzeichnis `/etc/xinetd.d` (siehe Listing 15.77):

```
service rsync
{
    disable          = no
    only_from       = 192.168.56.82
    socket_type     = stream
    wait            = no
    user            = root
    server           = /usr/bin/rsync
    server_args     = --daemon
    log_on_failure += USERID
}
```

Listing 15.77 Konfigurationsdatei für »rsync«

Sie sehen hier, dass nur der Rechner mit der IP-Adresse 192.168.56.82 auf den Dienst zugreifen kann. Das ist in unserem Fall der zweite Domaincontroller. Wollen Sie weitere Domain-

controller einrichten, können Sie die zusätzlichen IP-Adressen, durch Leerzeichen getrennt, in die Datei eintragen.

Im nächsten Schritt konfigurieren Sie den *rsync*-Server. Erstellen Sie für die Konfiguration die Datei */etc/rsyncd.conf* mit dem Inhalt aus Listing 15.78. Dabei spielt es keine Rolle, wie Sie den *rsync*-Server starten, die Konfigurationsdatei ist immer identisch:

```
[sysvol]
path = /var/lib/samba/sysvol/
comment = Samba sysvol
uid = root
gid = root
read only = yes
auth users = sysvol-repl
secrets file = /etc/samba/rsync.secret
```

Listing 15.78 Inhalt der Konfigurationsdatei des »rsyncd«

In die Datei */etc/samba/rsync.secret* tragen Sie den Benutzer ein, den Sie in dieser Datei als auth user registriert haben, sowie das Passwort, das dieser Benutzer verwenden soll. Ein Beispiel dafür sehen Sie in Listing 15.79:

```
sysvol-repl:geheim
```

Listing 15.79 Daten für die Authentifizierung

Der Benutzername und das Passwort werden durch einen Doppelpunkt getrennt. Setzen Sie die Rechte der Datei auf `root:root 600`. Starten Sie anschließend den *xinetd* neu. Unter Debian und Ubuntu sehen Sie in der Datei */var/log/syslog* die Meldungen wie in Listing 15.80:

```
root@adminbuch:~# journalctl -u xinetd
[...]
35:43 addc-01 xinetd[1099]: Starting internet superserver: xinetd.
Jan 03 15:35:43 addc-01 systemd[1]: Started LSB: Starts or stops the xinetd daemon..
[...]
Dez 28 15:35:43 addc-01 xinetd[1108]: Reading included configuration \
    file: /etc/xinetd.d/rsync [file=/etc/xinetd.d/rsync] [line=14]
[...]
Dez 28 15:35:43 addc-01 xinetd[1108]: 2.3.15.3 started with libwrap \
    loadavg labeled-networking options compiled in.
Dez 28 15:35:43 addc-01 xinetd[1108]: Started working: 1 available service
```

Listing 15.80 Auszug aus der Log-Datei

Hier sehen Sie, dass die Datei zur Konfiguration des *rsyncd* abgearbeitet wurde. Mit dem Kommando `ss` können Sie dann noch testen, ob der Port auch erreichbar ist (siehe Listing 15.81):

```
root@addc-01:~# ss -tlp | grep rsync
LISTEN 0 64 *:rsync *: * users:(("xinetd",pid=1108,fd=5))
```

Listing 15.81 Prüfung des Ports für »rsyncd«

Konfiguration aller anderen DCs

Auf allen weiteren DCs installieren Sie ebenfalls den *rsync*. Den *xinetd* benötigen Sie dort nicht, da es sich bei den DCs immer nur um einen *rsync*-Client handelt. Nach der Installation von *rsync* erstellen Sie eine Datei, in der nur das Passwort für den Zugriff auf den *rsync*-Server abgelegt wird.

In unserem Beispiel soll es die Datei */etc/samba/rsync.pass* sein. Achten Sie hier auch wieder darauf, dass die Rechte auf *root:root 600* stehen. Jetzt können Sie die Replikation testen. Verwenden Sie beim Testen auf jeden Fall den Parameter *--dry-run*. Damit verhindern Sie, dass die Replikation wirklich durchgeführt wird. Erst wenn das Ergebnis des Tests stimmt, sollten Sie die Replikation starten. In Listing 15.82 sehen Sie den Test der Replikation:

```
root@addc-02:/etc# rsync --dry-run -XAavz --delete-after --password-file=/etc/samba\
  /rsync.pass rsync://sysvol-repl@addc-01/sysvol/ /var/lib/samba/sysvol/
receiving file list ... done
./
example.net/
example.net/Policies/
example.net/Policies/25612555-3CC1-4D82-BBBF-E3E2DB54E9B4/
example.net/Policies/25612555-3CC1-4D82-BBBF-E3E2DB54E9B4/GPT.INI
example.net/Policies/25612555-3CC1-4D82-BBBF-E3E2DB54E9B4/Machine/
example.net/Policies/25612555-3CC1-4D82-BBBF-E3E2DB54E9B4/User/
example.net/Policies/25612555-3CC1-4D82-BBBF-E3E2DB54E9B4/User/Registry.pol
example.net/Policies/25612555-3CC1-4D82-BBBF-E3E2DB54E9B4/User/comment.cmtx
example.net/Policies/31B2F340-016D-11D2-945F-00C04FB984F9/
example.net/Policies/31B2F340-016D-11D2-945F-00C04FB984F9/GPT.INI
example.net/Policies/31B2F340-016D-11D2-945F-00C04FB984F9/MACHINE/
example.net/Policies/31B2F340-016D-11D2-945F-00C04FB984F9/USER/
example.net/Policies/6AC1786C-016F-11D2-945F-00C04FB984F9/
example.net/Policies/6AC1786C-016F-11D2-945F-00C04FB984F9/GPT.INI
example.net/Policies/6AC1786C-016F-11D2-945F-00C04FB984F9/MACHINE/
example.net/Policies/6AC1786C-016F-11D2-945F-00C04FB984F9/USER/
example.net/scripts/

sent 77 bytes received 2,101 bytes 1,452.00 bytes/sec
total size is 843 speedup is 0.39 (DRY RUN)
```

Listing 15.82 Test der Replikation

Hier sehen Sie, dass alle Gruppenrichtlinien und das Logon-Skript übertragen wurden.

**Achten Sie auf die Pfade**

Achten Sie darauf, dass die Pfade alle korrekt sind. Bei der Replikation werden später alle alten Einträge gelöscht und die neuen geschrieben. Stimmen hier die Pfade nicht, können Sie Ihr System unbrauchbar machen.

Jetzt können Sie den Parameter `--dry-run` aus der Befehlszeile entfernen und die erste Replikation durchführen. Prüfen Sie, ob alle Verzeichnisse und Dateien übertragen wurden. Wenn alle Dateien übertragen worden sind, können Sie mit dem nächsten Schritt fortfahren.

Einen Cronjob für die Replikation auf dem Client einrichten

Um die Replikation zu automatisieren, erstellen Sie als Benutzer `root` einen *Cronjob*, der regelmäßig die Replikation durchführt. Im Beispiel soll die Replikation alle fünfzehn Minuten vollzogen werden. In Listing 15.83 sehen Sie die Zeile für den *cron*:

```
* /15 * * * * rsync -XAavz --delete-after --password-file=/etc/samba/rsync.pass\  
rsync://sysvol-repl@adminbuch/sysvol/ /var/lib/samba/sysvol/
```

Listing 15.83 Eintrag in der »crontab«

Die Zeit für die Replikation ist immer davon abhängig, wie viele Änderungen Sie an den Gruppenrichtlinien und den Logon-Skripten vornehmen. Damit ist die Replikation des zweiten DC abgeschlossen. Wenn Sie noch weitere DCs in Ihrer Domäne haben, wiederholen Sie diesen Schritt auf allen weiteren DCs.



Schreiben Sie das Kommando in ein Shell-Skript, und rufen Sie über den Cron nur das Shell-Skript auf. Das hat den Vorteil, dass Sie später auch noch andere Aufgaben über das Skript steuern können und Änderungen einfacher durchzuführen sind.

Anpassen der »smb.conf« auf den Client-DCs

Auf den Client-DCs der Replikation setzen Sie die Freigaben `sysvol` und `netlogon` auf `read-only`. So verhindern Sie, dass Änderungen an dem Inhalt der Freigabe über das Netzwerk durchgeführt werden können. Passen Sie hierfür die Datei `/etc/samba/smb.conf` so an, wie in Listing 15.84 zu sehen ist:

```
[sysvol]  
path = /var/lib/samba/sysvol  
read only = yes  
  
[netlogon]  
path = /var/lib/samba/sysvol/example.net/scripts  
read only = yes
```

Listing 15.84 Anpassen der Freigabe »sysvol«

Wenn Sie jetzt eine neue Gruppenrichtlinie erstellen, achten Sie darauf, dass der Gruppenrichtlinieneditor immer auf den Domaincontroller mit der FSMO-Rolle *PdcEmulationMasterRole* zeigt. Das ist aber die Standardeinstellung, denn auch in einer Windows-Domäne werden die GPOs immer auf dem Domaincontroller mit der FSMO-Rolle *PdcEmulationMasterRole* geändert.

15.11 Was geht noch mit Samba 4?

Es ist unmöglich, den vollen Funktionsumfang von Samba 4 in einem Kapitel abzudecken, aber wir hoffen, dass Sie einen ersten Einblick in die Funktionen bekommen haben. Mit dieser kurzen Einführung sind Sie auf jeden Fall in der Lage, einen Samba 4 als Domaincontroller zu installieren und den Server für die Ausfallsicherheit zu replizieren.

Das Thema Gruppenrichtlinien konnte hier auch nur sehr kurz angeschnitten werden. Gerade das ist ein Bereich, bei dem es sich lohnt, tiefer einzusteigen. Mit den Gruppenrichtlinien können Sie nicht nur Berechtigungen vergeben oder Systemrechte entziehen, sondern Sie sind auch in der Lage, Drucker an Gruppen oder Abteilungen zuzuweisen. Sie können auch Netzwerklaufwerke über Gruppenrichtlinien verteilen. All das erfordert auf jeden Fall eine genaue Planung des Active Directory in Ihrem Netzwerk. Die Planung eines Active Directory ist eine komplexe Angelegenheit, die sehr stark von der Umgebung abhängig ist, in der es eingesetzt werden soll. Die Erläuterung der verschiedenen Vorgehensweisen kann hier nur sehr allgemein ausfallen Sie müssen Ihr Vorgehen unbedingt auf Ihre individuellen Anforderungen anpassen.

Natürlich ist auch eine Migration älterer Systeme wie Samba 3 und Windows-Server vom Windows Server 2000 bis zum Windows Server 2008 R2 hin zu Samba 4 möglich. Dafür sollten Sie sich die Informationen aus spezieller Literatur zum Thema Samba 4 oder aus dem Samba-Wiki unter https://wiki.samba.org/index.php/User_Documentation holen.

Kapitel 16

NFS

In Zeiten der zentralen Datenhaltung ist es besonders wichtig, dass Sie einen zuverlässigen Dienst haben, der für Ihre Anwender die benötigten Daten bereithält. Mit »NFSv4« steht Ihnen ein Dienst zur Verfügung, der Daten für Ihre Linux-Clients auf einem Server bereitstellen kann.

Mit dem *Network File System* (NFS) können Sie Festplattenpartitionen und Verzeichnisse von entfernten Systemen in Ihr lokales System einhängen. Dabei wird das entfernte Dateisystem im lokalen System direkt in den Dateibaum eingehängt. Die Aktion ist für den Anwender transparent, da – anders als unter Windows üblich – kein zusätzliches Netzlaufwerk in der Umgebung des Anwenders erscheint. Dadurch können Sie NFS sehr gut für die Bereitstellung von Daten auf Linux-Clients verwenden und die Daten zentral auf einem Server verwalten. Zusätzlich wird es für Sie einfach, die Nutzdaten der Anwender an einer zentralen Stelle zu sichern.

16.1 Unterschiede zwischen NFSv3 und NFSv4

Den aktuellen NFS-Server können Sie sowohl als NFSv3 als auch als NFSv4 einrichten. Der NFSv4-Server hat aber gegenüber dem NFSv3-Server folgende Vorteile:

- ▶ Das Filelocking wird direkt vom NFS-Server durchgeführt und nicht mehr durch einen separaten Daemon wie bei NFSv3. Gerade diese Funktion wird immer wichtiger, da Programme wie *LibreOffice* große Probleme mit dem Filelocking unter NFSv3 haben.
- ▶ Die Möglichkeiten des Cachings wurden deutlich verbessert. Dadurch werden die Zugriffe erheblich beschleunigt.
- ▶ Mit NFSv4 ist es möglich, eine Authentifizierung über *Kerberos* zu realisieren.
- ▶ Im Gegensatz zu NFSv3, bei dem alle Zugriffe grundsätzlich zustandslos sind, wird bei der Verbindung eines NFS-Clients mit dem NFSv4-Server eine Client-ID vom Server vergeben. Diese ID wird nach einer gewissen Zeit (*leasetime*) vom Server verworfen, wenn der Client sie nicht regelmäßig auffrischt. Sobald der NFS-Client eine Datei auf dem NFS-Server öffnet, erhält diese eine Zustands-ID (*State-ID*). Diese ID gibt dann Auskunft über die aktuelle Nutzung der Datei.

- ▶ Der NFSv4-Server kann die Dateioperationen an einen Client delegieren. Dadurch kann der Client die delegierten Dateien selbstständig im eigenen Cache verändern oder löschen. Diese Möglichkeit verringert den Netzwerkverkehr. Wenn nun der Server erkennt, dass ein anderer Client auf die Datei zugreifen will, kann der NFS-Server die Delegation widerrufen. Der Client schickt seine Änderungen dann umgehend zum Server. Diese Rücknahmen realisiert der NFS-Server über *Callback-RPCs*. Durch eine eventuell vorhandene Firewall zwischen dem Client und dem Server können die Rückrufe blockiert sein. Aus diesem Grund testet der NFS-Server diese Fähigkeit beim Verbindungsaufbau des Clients und kann dann das Verhalten entsprechend anpassen.
- ▶ Im Gegensatz zu NFSv3 kennt NFSv4 jetzt auch *UTF-8*, sodass es bei Dateinamen mit Sonderzeichen nicht mehr zu Problemen beim Speichern der Dateien kommt.
- ▶ Bei NFSv4 ist die Einschränkung auf maximal 16 Gruppen eines zugreifenden Benutzers aufgehoben. Bei NFSv4 sind bis zu 64 Gruppenmitgliedschaften eines Benutzers möglich. Die Anzahl hängt von dem Maximalwert ab, der beim Kompilieren des NFS-Servers eingetragen wurde. Längere Gruppenlisten sind auch nicht sinnvoll, da die Gruppenliste eines Benutzers beim Zugriff auf eine NFS-Freigabe als Integer-Array übergeben wird. Je länger das Array wird, umso aufwendiger wird die Suche nach der richtigen Gruppen-ID. Das würde die Performance des Servers erheblich beeinträchtigen.

Sie sehen, es besteht kein Grund mehr, NFSv3 einzusetzen. Auch das Umstellen eines NFSv3-Servers auf NFSv4 ist kein großes Hexenwerk. Der Aufwand lohnt auf jeden Fall, und Sie werden mit einem stabileren und schnelleren Netzwerkdateisystem belohnt.

16.2 Funktionsweise von NFSv4

Unter NFSv3 kommen immer noch mehrere einzelne Protokolle zum Einsatz. Diese regeln zusammen die Kommunikation zwischen dem NFS-Server und dem NFS-Client. Die einzelnen Protokolle bekommen vom *Portmapper* bei jedem Neustart einen Port zugewiesen. So müssen Sie neben dem eigentlichen NFS-Port (2049) und dem Port für den Portmapper (111) auch noch die Ports für die Protokolle *mountd*, *lockd* und *statd* auf einer eventuell vorhandenen Firewall freigeben.

Das Problem dabei ist: Die Ports werden dynamisch vom Portmapper vergeben und ändern sich bei jedem Neustart des Servers. Es gibt zwar eine Möglichkeit, diese Ports auch fest zu vergeben, aber da wir hier mit NFSv4 arbeiten wollen, werden wir auf diese Möglichkeit nicht mehr eingehen. Die verschiedenen Protokolle unter NFSv3 haben folgende Aufgaben:

- ▶ **mountd**
Das Mountprotokoll ist verantwortlich für den Zugriff auf die Freigaben.
- ▶ **lockd**
Das Lock-Protokoll setzt Dateisperren auf Dateien, die momentan in Verwendung sind.

► **statd**

Das Stat-Protokoll registriert die Dateien, die vom Client auf dem Server geöffnet sind.

Bei NFSv4 werden alle Protokolle, die unter NFSv3 noch eigenständig sind, zusammengefasst. Dadurch laufen bei NFSv4 alle Daten über einen Standardport, den TCP-Port 2049. Im Gegensatz zu NFSv3 ist TCP das Standardtransportprotokoll, bei NFSv3 war das noch UDP.

Eine weitere große Änderung in der Funktion von NFSv4 ist das Verwenden eines Pseudodateisystems, in das dann die eigentlichen Freigaben eingebunden werden. Die direkte Freigabe wie bei NFSv3 ist zwar auch noch möglich, wird aber nicht mehr empfohlen.

16.3 Einrichten des NFSv4-Servers

Bevor der NFS-Server konfiguriert werden kann, müssen die entsprechenden Pakete auf dem Server installiert werden. Bei Debian und Ubuntu sind das die Pakete `nfs-kernel-server` und `nfs-common`. Unter openSUSE und CentOS installieren Sie das Paket `nfs-kernel-server`.

Unter openSUSE aktivieren Sie zusätzlich noch den Dienst `nfsserver` im *Runleveleditor*.



16.3.1 Konfiguration des Pseudodateisystems

Bei NFSv4 wird der eigentliche Zugriff auf die Daten über ein Pseudodateisystem bereitgestellt. Der Vorteil ist der, dass ein Client nicht auf den Rest des Dateisystems zugreifen kann. Erzeugen Sie ein Verzeichnis, in das später alle Freigaben eingebunden werden. In Listing 16.1 sehen Sie, wie das Verzeichnis gleich mit den richtigen Rechten angelegt wird.

```
root@adminbuch:~# mkdir -m 1777 /nfs4root
```

Listing 16.1 Erzeugung des Mountpoints für das Pseudodateisystem

In unserem Beispiel sollen das Verzeichnis `/daten` und das Verzeichnis `/abteilungen` freigegeben werden. Für diese beiden Freigaben erzeugen Sie als Erstes jeweils einen `bind`-Mountpoint im Pseudodateisystem und setzen die benötigten Rechte. In Listing 16.2 sehen Sie das Erstellen der Mountpoints:

```
root@adminbuch:~# cd /nfs4root/
root@adminbuch:/nfs4root# mkdir -m 1777 daten
root@adminbuch:/nfs4root# mkdir -m 1777 abteilungen
```

Listing 16.2 Erzeugung der `bind`-Mountpoints

Jetzt können Sie die eigentlichen Verzeichnisse, in denen sich die Daten befinden, in die gerade erzeugten `bind`-Mountpoints einbinden.



Die Berechtigungen am bind-Mountpoint haben keine Auswirkungen auf die spätere Freigabe, sie dienen lediglich dazu, das eigentliche Verzeichnis an das Pseudodateisystem zu binden. Für die Zugriffe auf die Dateien gelten weiterhin die Berechtigungen in den Verzeichnissen der Freigabe (siehe Listing 16.3):

```
root@adminbuch:/nfs4root# mount --bind /daten daten/
root@adminbuch:/nfs4root# mount --bind /abteilungen abteilungen/
```

Listing 16.3 Einbinden der Datenverzeichnisse in die bind-Mountpoints

Wenn Sie das Einbinden ohne Fehler durchführen konnten, können Sie die bind-Mounts jetzt permanent in die Datei `/etc/fstab` eintragen:

```
/daten /nfs4root/daten none rw,bind 0 0
/abteilungen /nfs4root/abteilungen none rw,bind 0 0
```

Listing 16.4 Einträge in der Datei `»/etc/fstab«`

16.3.2 Anpassen der Datei `»/etc/exports«`

In der Datei `/etc/exports` werden die Freigaben für den NFS-Server eingetragen. Tragen Sie als Erstes das Pseudodateisystem in die Datei ein und anschließend die eigentlichen Freigaben:

```
# Eintrag für das Pseudodateisystem
/nfs4root 192.168.56.0/24(ro,sync,insecure,root_squash,no_subtree_check,fsid=0)

# Einträge für die Freigaben
/nfs4root/daten 192.168.56.0/24(rw,sync,insecure,root_squash,no_subtree_check)
/nfs4root/abteilungen 192.168.56.0/24(rw,sync,insecure,no_root_squash,\
no_subtree_check)
```

Listing 16.5 Einträge in der Datei `»/etc/exports«`

Die Parameter der Einträge haben die folgenden Bedeutungen:

- ▶ `/nfs4root`
Dabei handelt es sich um die hier verwaltete Freigabe. Im ersten Eintrag ist es das Pseudodateisystem, und in den beiden anderen Einträgen sehen Sie die eigentlichen Freigaben.
- ▶ `192.168.56.0/24`
Alle Hosts aus dem Netzwerk `192.168.56.0` haben Zugriff auf diese Freigaben. Über die Subnetzmaske können Sie den Zugriff bis auf einen Host genau beschränken.
- ▶ `ro`
Das Dateisystem wird `read-only` eingehängt. Da es sich hier um das Pseudodateisystem handelt, ist das die richtige Option, denn unter keinen Umständen soll ein Client direkt in das Pseudodateisystem schreiben dürfen. Dieser Parameter ist die Standardeinstellung.

▶ `rw`

Das Dateisystem wird `read-write` eingehängt. Setzen Sie diesen Parameter immer, wenn auf die Freigabe schreibend zugegriffen werden soll. Wenn Sie ihn nicht angeben, wird standardmäßig `read-only` (`ro`) gemountet.

▶ `sync/async`

Die Parameter `sync` und `async` verändern das Verhalten des Servers beim Speichern von Daten, die der Client an den Server sendet.

Hier sehen Sie das Verhalten bei der Einstellung `sync`:

1. Ein Programm möchte Daten auf die NFS-Freigabe schreiben.
2. Der Client leitet die Anfrage an den Server weiter.
3. Der Server leitet den Schreibzugriff an das lokale Dateisystem weiter.
4. Das lokale Dateisystem führt den Schreibzugriff aus.
5. Das lokale Dateisystem meldet dem Server, ob der Schreibzugriff erfolgreich war oder ob er fehlgeschlagen ist.
6. Der Server teilt dem NFS-Client mit, ob der Schreibzugriff erfolgreich war oder nicht.
7. Der Client teilt dem Programm mit, ob der Schreibzugriff erfolgreich war oder nicht.

Hier sehen Sie das Verhalten bei der Einstellung `async`:

1. Ein Programm möchte Daten auf die NFS-Freigabe schreiben.
2. Der Client leitet die Anfrage an den Server weiter.
3. Der Server teilt dem Client mit, dass das Schreiben erfolgreich war.
4. Der Client teilt dem Programm mit, dass das Schreiben erfolgreich war.
5. Der Server leitet den Schreibzugriff an das lokale Dateisystem weiter.
6. Das lokale Dateisystem führt den Schreibzugriff aus.

Da bei der Verwendung von `async` der Schreibvorgang schon vorzeitig als abgeschlossen an die Anwendung gesendet wird, kann es hier zu Datenverlusten kommen, wenn zum Beispiel die Festplatte voll ist.

▶ `secure/insecure`

Über diesen Parameter können Sie steuern, ob ein *NFS-Request* von einem TCP-Port kleiner 1024 stammen muss oder nicht. Die Voreinstellung ist `secure`. Um sie zu deaktivieren, wird `insecure` verwendet.

▶ `root_squash/no_root_squash`

Über diesen Parameter steuern Sie, welche Rechte der lokale Benutzer `root` auf dem Client an der Freigabe hat. Setzen Sie hier den Wert `root_squash`, erhält der lokale `root` auf der Freigabe nur die Rechte, die der Benutzer `nobody` auf dem Server hat. Nur wenn der lokale `root` volle Dateisystemrechte erhalten soll, setzen Sie hier den Wert auf `no_root_squash`. Die Standardeinstellung ist `root_squash`.

- ▶ `no_subtree_check/no_subtree_check`
Dieser Parameter schaltet das subtree-checking ein oder aus. Mit dem subtree-checking können Sie eine weitere Sicherheitsebene einführen. Dieser Parameter wird immer dann relevant, wenn Sie nur ein Verzeichnis freigegeben haben und nicht eine ganze Partition. Bei jedem Zugriff auf eine Freigabe muss der NFS-Server prüfen, ob sich die Datei, auf die der Client zugreifen will, in der Partition befindet. Zusätzlich muss er dann noch prüfen, ob sich die Datei auch in dem entsprechenden Verzeichniszweig befindet. Diese Prüfung ist etwas aufwendiger für den NFS-Server. Diese Prüfung wird als subtree-checking bezeichnet. Die Standardeinstellung ist `no_subtree_check`.
- ▶ `fsid=0`
Dieser Parameter wird nur für das Pseudodateisystem benötigt. Durch diesen Parameter weiß das System, dass es sich hierbei um das Pseudodateisystem des NFSv4-Servers handelt.

Nachdem Sie alle Änderungen an der Datei `/etc/exports` durchgeführt haben, laden Sie den NFS-Server neu. Bei allen Distributionen wird das Neuladen über das Kommando `systemctl` so wie in Listing 16.6 durchgeführt:

```
root@adminbuch:~# systemctl reload nfs-kernel-server
Re-exporting directories for NFS kernel daemon....
```

Listing 16.6 Neuladen der NFS-Serverkonfiguration bei Debian und Ubuntu



Es ist ratsam, an dieser Stelle wirklich nur ein `reload` durchzuführen und kein `restart`. Durch den Neustart des NFS-Servers würden die Clients kurzzeitig die Verbindung zum Server verlieren, und es könnte zu Datenverlusten kommen.

16.3.3 Tests für den NFS-Server

Nachdem Sie den NFS-Server gestartet haben, können Sie mit dem Kommando `rpcinfo -p localhost` den Server testen (siehe Listing 16.7). Dort sehen Sie dann alle NFS-Versionen, die der Server bereitstellen kann:

```
root@adminbuch:/nfs4root# rpcinfo -p localhost
  program vers proto  port  service
  100000    4   tcp    111   portmapper
  100000    3   tcp    111   portmapper
  100000    2   tcp    111   portmapper
  100000    4   udp    111   portmapper
  100000    3   udp    111   portmapper
  100000    2   udp    111   portmapper
  100005    1   udp   48615  mountd
  100005    1   tcp   42065  mountd
```

```

100005  2  udp  51865  mountd
100005  2  tcp  43881  mountd
100005  3  udp  60791  mountd
100005  3  tcp  53207  mountd
100003  3  tcp   2049  nfs
100003  4  tcp   2049  nfs
100227  3  tcp   2049
100003  3  udp   2049  nfs
100227  3  udp   2049
100021  1  udp  59451  nlockmgr
100021  3  udp  59451  nlockmgr
100021  4  udp  59451  nlockmgr
100021  1  tcp  37079  nlockmgr
100021  3  tcp  37079  nlockmgr
100021  4  tcp  37079  nlockmgr

```

Listing 16.7 Ausgabe des Kommandos »rpcinfo«

Hier sehen Sie, dass das Protokoll `mountd` nicht für die NFS-Version 4 vorhanden ist. Denn bei NFSv4 ist die Funktion direkt im NFS integriert.

Um zu testen, ob alle Freigaben auch richtig bereitgestellt werden, können Sie den Server mit dem Kommando `exportfs -v` so wie in Listing 16.8 prüfen:

```

root@adminbuch:~# exportfs -v
/nfs4root 192.168.123.0/24(ro,wdelay,insecure,root_squash,\
           no_subtree_check,fsid=0)

/nfs4root/daten 192.168.123.0/24(rw,wdelay,nohide,insecure,\
           root_squash,no_subtree_check)

/nfs4root/home 192.168.123.0/24(rw,wdelay,nohide,insecure,\
           no_root_squash,no_subtree_check)

```

Listing 16.8 Testen des NFS-Servers mit »exportfs -v«

Hier sehen Sie, dass ein weiterer Parameter beim Einrichten der Freigaben automatisch mit angegeben wird, nämlich der Parameter `wdelay`. Dieser Parameter sorgt dafür, dass die Schreibvorgänge etwas verzögert ausgeführt werden, wenn der Server mit weiteren Schreibzugriffen rechnet. Dadurch können mehrere Schreibvorgänge gemeinsam durchgeführt werden. Wenn Sie aber immer nur einzelne oder kurze Schreibzugriffe auf Ihren Server haben, können Sie die Standardeinstellung mit dem Parameter `no_wdelay` deaktivieren.

Der Parameter `no_wdelay` hat keinen Effekt, wenn Sie anstelle von `sync` den Parameter `async` verwenden.



16.4 Konfiguration des NFSv4-Clients

Damit der Client die Freigaben nutzen kann, benötigen Sie für jede der Freigaben ein Verzeichnis als Mountpoint auf dem Client. Um zu testen, ob sich die Freigaben ohne Fehler einbinden lassen, testen Sie das Mounten der Freigaben zunächst auf der Kommandozeile. Erst wenn der Test erfolgreich war, tragen Sie das Mounten dauerhaft in der Datei */etc/fstab* ein. Sollte es aufgrund fehlerhafter Einträge in der Datei */etc/fstab* beim Neustart des Systems zu Fehlern beim Mounten kommen, kann es Ihnen passieren, dass das System entweder gar nicht startet oder an der Stelle des Mountens sehr lange hängen bleibt. In Listing 16.9 sehen Sie das Mounten der Freigabe direkt über die Kommandozeile:

```
root@adminbuch-02:~# mount -t nfs4 adminbuch:/daten /daten
root@adminbuch-02:~# mount -t nfs4 adminbuch:/abteilungen /abteilungen
```

Listing 16.9 Mounten der Freigaben am Client

Wie Sie sehen, wird nicht der Pfad des Pseudodateisystems auf dem Server angegeben, sondern das eigentlich freigegebene Verzeichnis.



Wichtig ist hier, dass Sie die Option `-t nfs4` verwenden, da sonst der Client das NFSv3-Protokoll verwenden würde, was auf jeden Fall zu einer Fehlermeldung führt.

Sie können das gesamte Pseudodateisystem in einen Mountpoint einbinden, indem Sie das Wurzelverzeichnis des Pseudodateisystems beim Mounten angeben. Dadurch lassen sich freigegebene Ordner eines Servers zu einer Freigabe zusammenfassen. In Listing 16.10 sehen Sie beispielhaft, wie Sie das Wurzelverzeichnis des Pseudodateisystems einhängen:

```
root@adminrepl:~# mount -t nfs4 adminbuch:/ /alles
```

Listing 16.10 Mounten des Wurzelverzeichnisses des Pseudodateisystems

Wie Sie sehen, wird hier als Quelle einfach das Wurzelverzeichnis angegeben. Dadurch werden alle im Pseudodateisystem eingebundenen Verzeichnisse im lokalen Verzeichnis */alles* zusammen angezeigt. Im Anschluss daran können Sie das erfolgreiche Mounten der Freigabe noch mit dem Kommando `mount` testen. Das Ergebnis sehen Sie in Listing 16.11:

```
root@adminrepl:~# mount
[...]
adminbuch:/daten on /daten type nfs4 (rw,relatime,vers=4.2,\
  rsize=262144,wsiz=262144,namlen=255,\
  hard,proto=tcp,timeo=600,retrans=2,sec=sys,\
  clientaddr=192.168.56.85,local_lock=none,addr=192.168.56.81)
adminbuch:/ on /alles type nfs4 (rw,relatime,vers=4.2,\
  rsize=262144,wsiz=262144,namlen=255,\
  hard,proto=tcp,timeo=600,retrans=2,sec=sys,\
  clientaddr=192.168.56.85,local_lock=none,addr=192.168.56.81)
```

```
adminbuch:/abteilungen on /abteilungen type nfs4 (rw,relatime,vers=4.2,\
  rsize=262144,wsize=262144,namlen=255,\
  hard,proto=tcp,timeo=600,retrans=2,sec=sys,\
  clientaddr=192.168.56.85,local_lock=none,addr=192.168.56.81)
```

Listing 16.11 Prüfen des Mountens mit »mount«

Jetzt können Sie die Einträge in der Datei */etc/fstab* so wie in Listing 16.12 vornehmen:

```
# NFSv4 Mounts
adminbuch:/daten          /daten          nfs4  rw  0  0
adminbuch:/abteilungen   /abteilungen   nfs4  rw  0  0
adminbuch:/               /alles         nfs4  rw  0  0
```

Listing 16.12 Einträge in der Datei »/etc/fstab«

16.5 Konfiguration des idmapd

Anders als bei NFSv3 werden die UIDs der Benutzer bei NFSv4 nicht automatisch auf die Benutzernamen gemappt. Für das Mapping der Benutzernamen wird sowohl auf dem Client als auch auf dem Server der *idmapd* benötigt.

Wenn Sie Ihren NFS-Server und NFS-Client wie in den vorhergehenden Abschnitten konfiguriert und gemountet haben, sehen Sie bei einem `ls -l` auf dem Client die Ausgabe wie in Listing 16.13:

```
root@adminbuch-02:/daten# ls -l
insgesamt 12
drwxrwx--- 2 root nogroup 4096  8. Aug 12:16 abteilung
drwxrwx--- 2 root nogroup 4096  8. Aug 12:16 buchhaltung
drwxrwx--- 2 root nogroup 4096  8. Aug 12:13 verwaltung
```

Listing 16.13 Auflistung der Rechte ohne »idmapd«

Die Zuordnung der Originalverzeichnisse auf dem NFS-Server sieht aber so aus, wie in Listing 16.14 dargestellt:

```
root@adminbuch:/daten# ls -l
insgesamt 12
drwxrwx--- 2 root benutzer  4096  8. Aug 12:16 abteilung
drwxrwx--- 2 root buchhaltung 4096  8. Aug 12:16 buchhaltung
drwxrwx--- 2 root verwaltung  4096  8. Aug 12:13 verwaltung
```

Listing 16.14 Auflistung der Rechte auf dem Server

Jetzt kommt der *idmapd* ins Spiel. Über RPC-Aufrufe werden nun die UIDs der Benutzer und Gruppen auf die entsprechenden Namen gemappt. Wichtig ist an dieser Stelle, dass die

Benutzer und Gruppen sowohl auf dem Server als auch auf dem Client jeweils dieselbe UID besitzen. Das können Sie am einfachsten über eine zentrale Benutzerverwaltung realisieren. Der *idmapd* wird über die Konfigurationsdatei */etc/idmapd.conf* konfiguriert. In Listing 16.15 sehen Sie ein Beispiel für die Datei:

```
[General]
Verbosity = 0
Pipefs-Directory = /var/lib/nfs/rpc_pipefs
Domain = example.net

[Mapping]
Nobody-User = nobody
Nobody-Group = nogroup
```

Listing 16.15 Konfigurationsdatei des »idmapd«

Unter dem Abschnitt [General] müssen Sie lediglich Ihre DNS-Domäne eintragen. Das ist dann besonders wichtig, wenn Sie Kerberos zur Absicherung Ihres NFS-Servers einsetzen wollen. Unter dem Abschnitt [Mapping] sehen Sie die Zuordnung für anonyme Benutzer. Hier können Sie die Standardeinstellung belassen. Jetzt konfigurieren Sie Ihr System so, dass der *idmapd* auch automatisch gestartet wird. Bei Debian setzen Sie dazu auf dem Server und allen Clients in der Datei */etc/default/nfs-common* die Variable `NEED_IDMAPD=yes`. Anschließend starten Sie *nfs-common* mit dem Kommando `systemctl restart nfs-client` auf dem Server und den Clients neu.

Bei Ubuntu wird der *idmapd* immer gestartet, sobald Sie das Paket *nfs-common* installiert haben. Wichtig ist nur, dass Sie nach der Änderung an der Datei */etc/idmapd.conf* den Dienst mit dem Kommando `systemctl restart idmapd` neu starten.

Bei openSUSE und CentOS muss auf dem Server und den Clients in der Datei */etc/sysconfig/nfs* die Variable `NFS4_SUPPORT="yes"` gesetzt sein. Dadurch wird beim Starten des NFS-Servers oder des NFS-Clients auch automatisch der *idmapd* gestartet.

Nach dem Start des *idmapd* werden Ihnen mit `ls -l` auf dem Client auch wieder die richtigen Benutzer angezeigt, wie Sie in Listing 16.16 sehen. Damit ist die Konfiguration des NFSv4-Clients abgeschlossen. Aber es gibt noch ein paar Möglichkeiten, den Zugriff auf den Server zu optimieren.

```
root@adminbuch-02:/daten# ls -l
insgesamt 12
drwxrwx--- 2 root benutzer 4096 19. Sep 18:12 alle
drwxrwx--- 2 root buchhaltung 4096 8. Aug 12:16 buchhaltung
drwxrwx--- 2 root verwaltung 4096 19. Sep 18:18 verwaltung
```

Listing 16.16 Auflistung der Rechte mit »idmapd«

16.6 Optimierung von NFSv4

Optimierungen zum Datendurchsatz können sowohl auf der Server- als auch auf der Client-Seite durchgeführt werden. Bevor Sie aber an die Optimierung Ihres NFS-Servers gehen, prüfen Sie den momentanen Datendurchsatz, um Vergleichswerte zu erhalten.

16.6.1 Optimierung des NFSv4Servers

Beim NFS-Server gibt es die Möglichkeit, die Anzahl der gleichzeitig laufenden NFS-Prozesse zu verändern. Für Debian und Ubuntu wird in der Datei `/etc/default/nfs-kernel-server` die Standardeinstellung vorgenommen. Sie steht dort auf dem Wert `RPCNFSDCOUNT=8`. Bei openSUSE und CentOS finden Sie den Wert in der Datei `/etc/sysconfig/nfs`. Dort ist der Wert auf `USE_KERNEL_NFSD_NUMBER="4"` eingestellt. Die Anzahl der Prozesse sollte zwischen vier und acht Prozessen pro CPU-Kern liegen. Testen Sie, mit welchem Wert Sie die besten Ergebnisse erhalten. Nicht immer bringt ein maximaler Wert auch maximale Performance.

Die aktuelle Auslastung des NFS-Servers können Sie in der Datei `/proc/net/rpc/nfsd` sehen. Wichtig sind hier die letzten zehn Zahlen in der `th`-Zeile. In Listing 16.17 sehen Sie einen Auszug aus der Datei:

```
root@adminbuch:~# cat /proc/net/rpc/nfsd
rc 0 0 1
fh 0 0 0 0 0
io 0 0
th 8 0 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
ra 32 0 0 0 0 0 0 0 0 0 0 0 0
net 1 1 0 0
rpc 1 0 0 0 0
proc2 18 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
proc3 22 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
proc4 2 0 0
proc4ops 59 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Listing 16.17 Auszug aus der Datei `»/proc/net/rpc/nfsd«`

In der `th`-Zeile werden die folgenden Angaben gemacht:

- ▶ 8
Direkt nach dem `th` werden die maximal möglichen NFS-Threads angezeigt. Im Beispiel handelt es sich um einen Debian-Server mit der Standardeinstellung von 8 Threads.
- ▶ 0
Im Anschluss an die Anzahl der Threads steht der Wert, wie häufig bereits alle Threads gleichzeitig verwendet wurden. Wenn der Wert sehr hoch liegt, ist es sinnvoll, dass Sie die Anzahl der Threads erhöhen.

► 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

Die zehn Zahlen geben die Zeit in Sekunden an, in der alle Threads in Benutzung waren. Die erste Zahl gibt einen Wert von 1 bis 10 % an, die zweite 11 bis 20 %, bis zur letzten Zahl 91 bis 100 %. Je höher die Werte, gerade im oberen Bereich zwischen 70 und 100 %, liegen, umso langsamer werden die Zugriffe der Clients.

Kontrollieren Sie diese Werte regelmäßig, damit eine zu geringe Anzahl an NFS-Threads nicht zum Flaschenhals Ihres Systems wird.

16.6.2 Optimierung des NFSv4-Clients

Beim NFS-Client werden alle Optimierungen über die Einträge in der Datei */etc/fstab* durchgeführt. Mit den Parametern *rsize* und *wsize* lassen sich die Puffergrößen für Lese- und Schreiboperationen anpassen. Testen Sie, mit welchem Wert Sie die besten Ergebnisse erhalten. Der maximale Wert bei NFSv4 kann 1.048.576 betragen.

Am Anfang sollte zunächst der Durchsatz ohne veränderte Parameter getestet werden. Dazu können Sie das folgende Kommando aus Listing 16.18 verwenden:

```
root@adminbuch-02:~# time dd if=/dev/zero of=/daten/testdatei bs=32k count=16384
```

Listing 16.18 Test ohne veränderte Parameter

Bei dem Test wird eine 512 MB große Datei auf der NFS-Freigabe erstellt. Um den Wert aussagekräftiger zu machen, wiederholen Sie den Test mehrfach. Die Datei sollte mindestens zweimal so groß sein wie der vorhandene Arbeitsspeicher des NFS-Servers.

Im zweiten Test setzen Sie jetzt den Wert in der */etc/fstab* auf einen mittleren Wert von 16.384, und zwar sowohl für *rsize* als auch für *wsize*. Die Zeile in der */etc/fstab* sieht dann so aus wie in Listing 16.19:

```
#adminbuch:/daten /daten nfs4 rw,rsize=16384,wsiz=16384 0 0
```

Listing 16.19 »/etc/fstab« mit Anpassungen für »rsize« und »wsiz«

Nach der Änderung muss die Freigabe vor dem erneuten Test neu gemountet werden, da sonst die neuen Werte nicht wirksam sind. Auch diesen Test wiederholen Sie mehrfach. Um ein optimales Ergebnis zu erzielen, wiederholen Sie den Test mit unterschiedlichen Werten. Natürlich können Sie alle zusätzlichen Parameter beim Mounten verwenden, die auch für andere Dateisysteme benutzt werden können. Mehr dazu finden Sie in den Manpages zu *nfs* und *fstab*. Um den besten Wert für den Parameter *rsize* zu finden, können Sie auch hier einen Test mit dem Kommando *dd* durchführen:

```
root@adminrepl:~# time dd if=/daten/testdatei of=/dev/null bs=32k
```

Listing 16.20 Test zur Ermittlung der Lesegeschwindigkeit

16.7 NFSv4 und Firewalls

Bei NFSv3 gibt es immer das Problem mit den dynamischen Ports für die Daemons *lockd*, *statd* und *mountd*, die bei jedem Start des Servers über den Portmapper neu vergeben werden. Bei NFSv4 werden die Funktionen dieser Daemons direkt vom NFS-Daemon übernommen. Dadurch wird der Portmapper nicht mehr benötigt. Bei einem reinen NFS-Server würde das folgende Firewallskript aus Listing 16.21 ausreichen, um nur noch den Zugriff über NFS und *ssh* zu erlauben:

```
#!/bin/bash
IPT=/sbin/iptables

$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP
$IPT -F

$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

$IPT -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
$IPT -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
# ssh erlauben
$IPT -A INPUT -m state --state NEW -p tcp --dport ssh -j ACCEPT

# NFS erlauben
$IPT -A INPUT -m state --state NEW -p tcp --dport nfs -j ACCEPT
```

Listing 16.21 Firewallskript für einen NFS-Server

Ein Test mit dem Portscanner *nmap* liefert das Ergebnis aus Listing 16.22. Der Zugriff auf die NFS-Freigaben ist nach wie vor möglich.

```
root@adminbuch-02:~# nmap -PO 192.168.56.81

Starting Nmap 5.00 ( http://nmap.org ) at 2022-12-12 15:30 CEST
Interesting ports on 192.168.56.81:
Not shown: 996 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
2049/tcp  open  nfs
```

Nmap done: 1 IP address (1 host up) scanned in 4.85 seconds

Listing 16.22 Ausgabe von »nmap«

16.8 NFS und Kerberos

Mit NFSv4 ist es möglich, eine sichere Authentifizierung über Kerberos zu realisieren. Wie Sie einen Kerberos-Server einrichten und verwalten, können Sie in Kapitel 14, »Kerberos«, nachlesen. An dieser Stelle gehen wir davon aus, dass Sie bereits einen Kerberos-Server installiert haben und nur noch die Änderungen für den NFS-Server nachtragen müssen.

Warum sollten Sie Ihren NFS-Server über Kerberos absichern? Sehen wir uns dazu kurz an, wie NFS arbeitet, wenn es keine Authentifizierung über Kerberos gibt.

Ohne Kerberos wird `AUTH_SYS` für die Authentifizierung verwendet. Zwei Hosts, der Server und der Client, besitzen dieselben UIDs und GIDs, die auf unterschiedlichem Wege bereitgestellt werden können. In kleinen Netzen haben Sie eventuell alle Benutzer mit derselben UID lokal angelegt, oder aber Sie verwenden LDAP für eine zentrale Benutzerverwaltung.

Wenn jetzt ein Host eine Anfrage per *Remote Procedure Call* (RPC) stellt, wird der Benutzer über die UID identifiziert. Da RPC weder signiert noch über eine Authentifizierung abgesichert ist, kann der Server nicht feststellen, ob die Anfrage von einem autorisierten Benutzer gestellt wird.

Durch den Einsatz von Kerberos können Sie die RPCs jetzt aber über Tickets absichern. Dafür stellt die NFS-Version 4 den Mechanismus *RPCSEC GSSAPI* zur Verfügung. GSSAPI ist lediglich eine Schnittstelle, über die verschiedene Sicherheitsmechanismen an einen Dienst gebunden werden können. In diesem Fall soll Kerberos mit NFS verbunden werden. Eine sehr gute Beschreibung von GSSAPI finden Sie unter <http://de.wikipedia.org/wiki/GSSAPI>.

16.8.1 Erstellung der Principals und der keytab-Dateien

Für alle NFS-Server und NFS-Clients erstellen Sie als Erstes einen *Principal* und eine *keytab*-Datei auf dem Kerberos-Server.

Die *keytab*-Datei des NFS-Servers stellen Sie auf dem NFS-Server bereit, und die *keytab*-Datei der NFS-Clients benötigen Sie auf den NFS-Clients. Die Principals und *keytab*-Dateien werden mit dem Programm `kadmin` erzeugt (siehe Listing 16.23):

```
kadmin.local: addprinc -randkey host/adminbuch.example.net
WARNUNG: Für host/adminbuch.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwandt.
Principal "host/adminbuch.example.net@EXAMPLE.NET" created.

kadmin.local: addprinc -randkey host/adminbuch-02.example.net
WARNUNG: Für host/adminbuch-02.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwandt.
Principal "host/adminbuch-02.example.net@EXAMPLE.NET" created.
```

```
kadmin: addprinc -randkey nfs/adminbuch.example.net
WARNUNG: Für nfs/adminbuch.example.net@EXAMPLE.NET wurde keine Richtlinie \
angegeben, es wird die Vorgabe »keine Richtlinie« verwandt.\
Principal "nfs/adminbuch.example.net@EXAMPLE.NET" created.
```

```
kadmin: ktadd -k /root/nfs.keytab nfs/adminbuch.example.net
Der Eintrag für Principal nfs/adminbuch.example.net mit KVNO 2 \
und Verschlüsselungstyp aes256-cts-hmac-sha1-96 wurde der \
Schlüsseltabelle WRFILE:/root/nfs.keytab hinzugefügt.
Der Eintrag für Principal nfs/adminbuch.example.net mit KVNO 2 \
und Verschlüsselungstyp aes128-cts-hmac-sha1-96 wurde der \
Schlüsseltabelle WRFILE:/root/nfs.keytab hinzugefügt.
```

```
kadmin: ktadd -k /root/adminbuch.keytab host/adminbuch.example.net
Der Eintrag für Principal host/adminbuch.example.net mit KVNO 2 \
und Verschlüsselungstyp aes256-cts-hmac-sha1-96 wurde der \
Schlüsseltabelle WRFILE:/root/adminbuch.keytab hinzugefügt.
Der Eintrag für Principal host/adminbuch.example.net mit KVNO 2 \
und Verschlüsselungstyp aes128-cts-hmac-sha1-96 wurde der \
Schlüsseltabelle WRFILE:/root/adminbuch.keytab hinzugefügt.
```

```
kadmin: ktadd -k /root/adminbuch-02.keytab host/adminbuch-02.example.net
Der Eintrag für Principal host/adminbuch-02.example.net mit KVNO 2 \
und Verschlüsselungstyp aes256-cts-hmac-sha1-96 wurde der \
Schlüsseltabelle WRFILE:/root/adminbuch-02.keytab hinzugefügt.
Der Eintrag für Principal host/adminbuch-02.example.net mit KVNO 2 \
und Verschlüsselungstyp aes128-cts-hmac-sha1-96 wurde der \
Schlüsseltabelle WRFILE:/root/adminbuch-02.keytab hinzugefügt.
```

Listing 16.23 Erzeugung der Principals und der »keytab«-Dateien

Kopieren Sie die Host-keytab-Dateien auf den NFS-Server und die Clients, und zwar als */etc/krb5.keytab*. Wenn Sie schon eine *krb5.keytab* für einen Host erstellt haben, benötigen Sie keine neue Datei.



Der NFS-Server verwendet die */etc/krb5.keytab* zur Authentifizierung. Aus diesem Grund werden jetzt die beiden keytab-Dateien zusammengeführt. Dazu verwenden Sie das Programm *ktutil*. In Listing 16.24 werden die beiden zuvor erstellten Clientdateien zusammengeführt. Damit sich der NFS-Client am NFS-Server authentifizieren kann, wird der Schlüssel für den NFS-Service auf dem NFS-Client in die *krb5.keytab*-Dateien jedes Clients eingetragen.

```
ktutil: rkt /etc/krb5.keytab
ktutil: rkt /root/nfs.keytab
```

```

ktutil: l
slot KVNO Principal
-----
  1  3  host/adminbuch.example.net@EXAMPLE.NET
  2  3  host/adminbuch.example.net@EXAMPLE.NET
  3  2  nfs/adminbuch.example.net@EXAMPLE.NET
  4  2  nfs/adminbuch.example.net@EXAMPLE.NET
ktutil: wkt /etc/krb5.keytab
ktutil: q

```

Listing 16.24 Zusammenführen zweier »keytab«-Dateien

16.8.2 Kerberos-Authentifizierung unter Debian und Ubuntu

Damit der NFS-Server überhaupt eine Kerberos-Authentifizierung durchführen kann, setzen Sie auf dem NFS-Server in der Datei `/etc/default/nfs-kernel-server` die Variable `NEED_SVCGSSD` auf `yes`. Anschließend starten Sie den NFS-Server neu.



»idmapd« konfigurieren

An dieser Stelle ist es sehr wichtig, dass Sie in der Datei `/etc/idmapd.conf` den Eintrag `Domain = example.net` auf Ihre Domain einstellen.

16.8.3 Kerberos-Authentifizierung auf openSUSE und CentOS

Auch bei openSUSE sorgen Sie erst einmal dafür, dass der NFS-Server die Kerberos-Authentifizierung überhaupt durchführt.

Dazu setzen Sie in der Datei `/etc/sysconfig/nfs` die Variable `NFS_SECURITY_GSS="yes"` und starten anschließend den NFS-Server neu.

16.8.4 Anpassen der Datei »/etc/exports«

Im nächsten Schritt geht es darum, die Datei `/etc/exports` anzupassen. In Listing 16.25 sehen Sie die neue Datei `/etc/exports`:

```

# /etc/exports: the access control list for filesystems which may be exported
#                to NFS clients.  See exports(5).


/nfs4root gss/krb5i(ro,sync,insecure,root_squash,no_subtree_check,fsid=0)
/nfs4root/daten gss/krb5i(rw,insecure,no_root_squash,no_subtree_check)

```

Listing 16.25 Die Datei `/etc/exports` mit Kerberos

Dort haben Sie drei verschiedene Möglichkeiten, das Sicherheitslevel einzustellen:

1. `gss/krb5`
Bei dieser Einstellung wird die Identität des Benutzers überprüft. Es findet keine Signatur der RPCs und auch keine Verschlüsselung der RPCs statt.
2. `gss/krb5i`
Bei dieser Einstellung werden die RPCs signiert. Dadurch wird verhindert, dass RPCs verändert werden können. Eine Verschlüsselung findet aber auch noch nicht statt.
3. `gss/krb5p`
Jetzt werden die RPCs auch zusätzlich verschlüsselt und können nur noch vom NFS-Server und vom NFS-Client gelesen werden.

Die Höhe der von Ihnen gewählten Sicherheit macht sich auch im Datendurchsatz bemerkbar. Testen Sie, welche Sicherheitsstufe für Sie die beste ist. 

Zum Abschluss der Konfiguration machen Sie die Änderungen noch bekannt. Dazu laden Sie die Konfiguration des NFS-Servers neu.

16.8.5 Einen NFS-Client für Kerberos unter Debian und Ubuntu konfigurieren

Auf der Clientseite setzen Sie jetzt in der Datei `/etc/default/nfs-common` die Variable `NEED_GSSD=yes`. Nachdem Sie die Datei geändert haben, starten Sie unter Debian den `nfs-client` mit dem Kommando `systemctl restart nfs-client` neu. Auf einem Ubuntu-System starten Sie den `gssd` mit dem Kommando `systemctl start gssd`.

16.8.6 Einen NFS-Client für Kerberos unter openSUSE und CentOS konfigurieren

Für den NFS-Client setzen Sie, genau wie schon vorher für den NFS-Server, in der Datei `/etc/sysconfig/nfs` die Variable `NFS_SECURITY_GSS="yes"` und starten dann den NFS-Dienst mit dem Kommando `rcnfs restart` neu.

16.8.7 Testen der durch Kerberos abgesicherten NFS-Verbindung

Jetzt können Sie den ersten Test durchführen. In Listing 16.26 sehen Sie den Mountbefehl und das anschließende Ergebnis:

```
root@adminbuch-02:~# mount -t nfs4 -o sec=krb5i adminbuch:/daten /daten
root@adminbuch-02:~# mount
[...]
adminbuch:/daten on /daten type nfs4 (rw,sec=krb5i,addr=192.168.56.81,\
clientaddr=192.168.56.85)
```

Listing 16.26 NFS-Mount mit Kerberos

Auf dem NFS-Server können Sie die Verbindung mit dem Kommando `exportfs -v` so wie in Listing 16.27 prüfen:

```
root@adminbuch:~# exportfs -v
/nfs4root          gss/krb5i(ro,wdelay,insecure,root_squash,no_subtree_check,fsid=0)
/nfs4root/daten    gss/krb5i(rw,wdelay,insecure,no_root_squash,no_subtree_check)
```

Listing 16.27 Prüfung auf dem NFS-Server



Auf einem Debian-System haben Sie die Möglichkeit, ein Debug-Level für die Kerberos-Authentifizierung zu setzen. Dazu setzen Sie auf der Clientseite in der Datei `/etc/default/nfs-common` die Variable `RPCSSDOPTS=vvv -rrr`. Nach der Fehlersuche sollten Sie diesen Parameter auf jeden Fall wieder auskommentieren und `nfs-common` neu starten, da sonst die Log-Datei schnell vollgeschrieben wird.

Nachdem Sie alles konfiguriert und getestet haben, fehlt jetzt nur noch der Eintrag in der Datei `/etc/fstab` auf dem Client, damit die Freigaben bei jedem Neustart auch automatisch gemountet werden. In Listing 16.28 sehen Sie den Eintrag auf dem Client:

```
adminbuch:/daten    /daten nfs4    defaults,sec=krb5i    0    0
```

Listing 16.28 Die Datei `»/etc/fstab«`

16.8.8 Testen der Verbindung

Wenn sich jetzt ein Benutzer auf dem NFS-Client anmeldet, erhält er zunächst nur ein TGT vom Kerberos-Server.

In Listing 16.29 sehen Sie eine `ssh`-Anmeldung an einem NFS-Client (192.168.123.191) und das Auflisten des TGT:

```
stefan@stefan~> ssh ktom@192.168.56.85
ptau@192.168.56.85's password:
```

```
ktom@adminbuch-02:~# klist
Ticketzwischenspeicher: FILE:/tmp/krb5cc_0
Standard-Principal: ktom@EXAMPLE.NET
```

```
Valid starting      Expires             Service principal
27.12.2022 14:07:22  28.12.2022 00:07:22  krbtgt/EXAMPLE.NET@EXAMPLE.NET
    erneuern bis 28.12.2022 14:07:20
```

Listing 16.29 `»ssh«`-Anmeldung am NFS-Client

Erst wenn der Anwender das erste Mal auf eine NFS-Freigabe zugreift, erhält er ein Ticket vom TGS. Ein Beispiel dazu sehen Sie in Listing 16.30:


```
charly@nfs-client:~$ cd /daten/  
charly@nfs-client:/daten$ klist  
Ticket cache: FILE:/tmp/krb5cc_0  
Default principal: ktom@EXAMPLE.NET
```

```
Valid starting    Expires          Service principal  
27.12.2022 14:07:22 28.12.2022 00:07:22 krbtgt/EXAMPLE.NET@EXAMPLE.NET  
    erneuern bis 28.12.2022 14:07:20  
27.12.2022 14:10:02 28.12.2022 00:10:02 nfs/adminbuch.example.net@EXAMPLE.NET  
    erneuern bis 28.12.2022 14:10:02
```

Listing 16.30 Zugriff auf eine NFS-Freigabe mit Kerberos-Authentifizierung

Das NFS-Ticket wird dem Benutzer angezeigt, da er den Dienst angefordert hat. Selbst wenn Sie mit `kdestroy` das Ticket löschen, bleibt die Verbindung zum NFS-Server bestehen.

Jetzt können Sie NFSv4 mit einer gesicherten Authentifizierung über Kerberos in Ihrem Netzwerk einsetzen.

Kapitel 17

LDAP

Verzeichnisdienste wie OpenLDAP ermöglichen es Ihnen, die Verwaltung der Ressourcen zentral zu steuern und an mehrere Stellen zu replizieren. Je größer Ihr Netzwerk wird, umso wichtiger werden die effiziente Verwaltung und Kontrolle aller Ressourcen. Mit OpenLDAP haben Sie einen Dienst, der Ihnen die tägliche Arbeit sehr stark erleichtern kann, da alle Ressourcen in einer hierarchischen Struktur abgelegt werden, die der Struktur Ihres Unternehmens entspricht. In diesem Kapitel lernen Sie OpenLDAP intensiv kennen.

In Zeiten, in denen die Netzwerkinfrastrukturen in Unternehmen immer heterogener werden, ist es besonders wichtig, dass Sie eine zentrale Verwaltung der Ressourcen im Netzwerk realisieren können. Unterschiedliche Netzwerkdienste, die den Benutzern zur Verfügung gestellt werden, verlangen oftmals eine eigene Benutzerverwaltung. Auf die Dauer kann es für Sie sehr schwierig und unübersichtlich werden, die verschiedenen Accounts zu organisieren. Das gilt nicht nur für Sie als Administrator, sondern auch für die Benutzer, die diese verschiedenen Dienste nutzen möchten.

Durch den Einsatz von LDAP werden die Benutzer und Ressourcen zentral verwaltet. Für jeden Benutzer existiert anschließend nur noch ein Konto. Mit diesem Konto kann sich der Benutzer an den verschiedensten Diensten anmelden. Ändert ein Benutzer sein Passwort, gilt das neue Passwort sofort für alle Dienste. Zusammen mit Kerberos können Sie zusätzlich ein *Single Sign-on* realisieren. Um diese zentrale Verwaltung der Ressourcen realisieren zu können, wurde das *Lightweight Directory Access Protocol* (LDAP) entwickelt. Bei LDAP handelt es sich ursprünglich um ein Protokoll, das als »Proxy« für den Zugriff auf eine *DAP-Datenbank* diente. Später wurde daraus ein eigenständiger Verzeichnisdienst.

LDAP ist ein Verzeichnisdienst, der hierarchisch gegliedert wird. Dadurch können Sie Ihre Unternehmensstruktur direkt im LDAP abbilden. Sie können aber genauso eine Struktur erstellen, die verschiedene Ressourcen zusammenfasst – eine allgemeingültige Struktur für einen LDAP-Baum gibt es nicht. Sie müssen immer Ihre Struktur genau an Ihre Gegebenheiten anpassen. Eine umfangreiche und ausführliche Planung kann Ihnen später eine Menge Arbeit ersparen. In diesem Kapitel wird ein OpenLDAP-Server konfiguriert werden, und wir zeigen Ihnen am Beispiel einiger Dienste, wie Sie OpenLDAP für eine zentrale Verwaltung Ihrer Ressourcen verwenden können. Zusätzlich wird hier auch die Einrichtung einer verschlüsselten Verbindung mit TLS implementiert. Ein weiteres wichtiges Thema, das hier

angesprochen wird, ist die Realisierung der Ausfallsicherheit durch zwei weitere OpenLDAP-Server in Ihrem Netzwerk.

In dieser Auflage des Buches werden wir auf die neue Version von OpenLDAP eingehen. Nach über 14 Jahren ist mit der Version 2.5 Anfang 2022 eine grundlegende Neuerung erschienen, mit der die Unterstützung für die alte Version 2.4 komplett eingestellt wurde. Es werden nicht einmal mehr Sicherheitsupdates bereitgestellt. Sie sollten also unbedingt auf die neue Version wechseln. Im Gegensatz zu den vorherigen Auflagen des Buches werden wir eine Multi-Provider-Replikation einrichten, sodass Sie im Anschluss mehrere beschreibbare Provider nutzen können. Auch werden wir auf die Replikation der Konfiguration eingehen, die seit der Version 2.5 funktioniert.

Neben der Version 2.5, bei der es sich um die LTS-Version des OpenLDAP handelt, wird auch eine Version 2.6 angeboten. In der Version 2.6 werden immer die neuesten Änderungen im Funktionsumfang bereitgestellt. Wenn es sich anbietet, werden wir auch auf die Version 2.6 eingehen. Leider gibt es momentan noch keine Distribution, die aktuelle Pakete der neuen Version bereitstellt. Aus diesem Grund werden wir für die Beispiele die Pakete der Firma Symas (www.symas.com) einsetzen. Diese Pakete gibt es für verschiedene Distributionen. Wir werden Ihnen die Besonderheiten der Pakete zeigen und auf die Installation näher eingehen.

17.1 Einige Grundlagen zu LDAP

Das *Lightweight Directory Access Protocol* (LDAP) wurde im Jahre 1993 entwickelt, um den Zugriff auf DAP-Datenbanken über TCP/IP zu erleichtern. Der ursprüngliche X.500-Standard, der für DAP-Datenbanken verwendet wurde, umfasst alle sieben Ebenen des OSI-Referenzmodells, wodurch eine Implementation über verschiedene Systeme hinweg recht aufwendig war. Deshalb wurde im Jahre 1993 das Protokoll LDAP entwickelt. Zunächst wurde es nur dazu verwendet, um auf DAP-Server zugreifen zu können. LDAP diente dabei mehr oder weniger als Proxy, um zwischen X.500 und den verschiedenen Systemen zu vermitteln. Der große Vorteil von LDAP gegenüber einer reinen DAP-Umgebung ist, dass für LDAP nur ein funktionsfähiger TCP/IP-Protokollstack benötigt wird. Später wurde zu LDAP ein eigenes Datenbank-Backend hinzugefügt, um unabhängig von den DAP-Servern zu werden.

17.1.1 Was ist ein Verzeichnisdienst?

In einem Verzeichnisdienst werden Informationen in einer hierarchischen Struktur abgelegt, die auch als baumartige Struktur bezeichnet wird. Die Vorteile einer solchen Struktur sind:

- ▶ Die Administration kann auf verschiedene Bereiche aufgeteilt werden.
- ▶ Die gesamte Unternehmensstruktur kann 1:1 im LDAP-Baum abgebildet werden.
- ▶ Durch eine Partitionierung kann die Struktur auf mehrere Server verteilt werden.

- ▶ Für die Suche nach bestimmten Objekten im LDAP-Baum können mehr oder weniger komplexe Filter eingesetzt werden, die die Suche auf bestimmte Teilbereiche des LDAP-Baums einschränken und dadurch die Suche beschleunigen.
- ▶ Da LDAP auf X.500 basiert, werden hier objektorientierte Datenmodelle verwendet. Dadurch ist eine Vererbung von Eigenschaften auf andere Objekte möglich.

Für Verzeichnisdienste existiert eine Reihe von X.500-Standards. Diese Standards beschreiben, wie Verzeichnisdaten zur Verfügung gestellt und abgerufen werden und wie die Verschlüsselung, Authentifizierung, Replikation und Verwaltung der Verzeichnisdaten gehandhabt werden. Die X.500-Standards liefern die Funktionsmodelle und Begriffsbestimmungen für die Verzeichnisdienste, die nicht voll auf dem X.500-Standard basieren, was bei LDAP der Fall ist. Die aktuelle Version von LDAP ist die Version 3. Diese bietet gegenüber der alten LDAP-Version 2 einige Vorteile:

- ▶ Authentifizierung durch Verwendung von SASL für die Verschlüsselung der Passwörter
- ▶ Verschlüsselung des gesamten Datenverkehrs im Netz durch TLS
- ▶ Möglichkeit der Verwendung von UTF-8 für Attribute
- ▶ Verweise auf andere LDAP-Server, die *Referrals*. Dadurch können Ressourcen in mehreren Bäumen gemeinsam genutzt, aber getrennt administriert werden.

LDAPv3 ist nicht abwärtskompatibel und sollte nicht mit LDAPv2-Servern in einer Umgebung betrieben werden.



17.1.2 Der Einsatz von LDAP im Netzwerk

Sie können LDAP auf verschiedene Arten und Weisen in Ihrem Netzwerk für die Verwaltung einsetzen. Auch ist LDAP die Grundlage anderer Verzeichnisdienste, wie zum Beispiel des Novell eDirectory (früher Novell NDS) oder des Microsoft Active Directory. Mithilfe von LDAP können Sie in Netzwerken die Verwaltung der Ressourcen vereinfachen. Zusammen mit Kerberos ist es auch möglich, ein *Single Sign-on* im Netzwerk zu realisieren.

LDAP lässt sich für eine Vielzahl von Diensten zur Authentifizierung und zur Bereitstellung von Ressourcen nutzen. Nach der Konfiguration des LDAP-Servers soll die Anbindung der folgenden Dienste realisiert werden, wobei es hier nicht um die vollständige Konfiguration der einzelnen Dienste geht, sondern nur um die Anbindung an LDAP:

- ▶ Verwaltung von Benutzern und Gruppen für POSIX-Konten
- ▶ Verwaltung von Benutzern und Gruppen für Samba-Konten
- ▶ Verwaltung von Weiterleitungen/Aliassen für den Postfix-Mailserver
- ▶ Einrichtung der Benutzerauthentifizierung für den IMAP-Server *Dovecot*
- ▶ Authentifizierung von Benutzern für die Anmeldung am Proxy *Squid*

- ▶ Authentifizierung von Benutzern für die Anmeldung am Apache-Webserver
- ▶ Zwei-Faktor-Authentifizierung mittels TOTP (ab Version 2.5)
- ▶ automatische Bereitstellung von Clientzertifikaten über AutoCA ab der Version 2.5

17.1.3 Aufbau des LDAP-Datenmodells

Wie schon in der Einleitung angesprochen wurde, ist das Datenmodell von LDAP objektorientiert. Bei LDAP gelten fast die gleichen Regeln wie bei der objektorientierten Programmierung. Auch im LDAP-Baum gibt es Objekte, Klassen, Vererbung und Polymorphie. Die Aufgabe von LDAP ist es, die Objekte abzubilden und miteinander in Beziehung zu bringen. Ein Objekt wird im LDAP-Baum als *Verzeichniseintrag* bezeichnet. Jedes Objekt wird über seinen eindeutigen Namen, den *Distinguished Name* (DN), im *Directory Information Tree* (DIT) angelegt, der ähnlich wie der Name einer Datei im Dateisystem behandelt wird. Durch die verschiedenen Objekte entsteht so nach und nach eine Baumstruktur.

Im LDAP-Baum wird zwischen zwei Objektarten unterschieden. Zunächst gibt es die *Organizational Unit* (OU), bei der es sich um ein Containerobjekt handelt. In diesen Organisational Units können weitere Objekte erzeugt werden. Die OUs werden zum Aufbau der Struktur verwendet. Die andere Objektart sind die Blattobjekte, die zum Beispiel mit der Kennzeichnung *Common Name* (CN) und *Users ID* (UID) im DIT Verwendung finden. Diese Objekte dienen zur Verwaltung der Ressourcen. Die Bezeichnungen werden aus den Objektklassen und Schemata abgeleitet, auf die im Verlauf dieses Kapitels noch weiter eingegangen wird.



OpenLDAP weist eine Besonderheit gegenüber dem NDS oder dem AD auf. Im OpenLDAP-Baum können Sie unterhalb des Blattobjekts weitere Objekte erzeugen. So ist es möglich, unterhalb eines Benutzers eine weitere OU zu erzeugen, in der dann zum Beispiel das persönliche Adressbuch des Benutzers gespeichert werden kann.

17.1.4 Objekte

Ein Objekt im LDAP-Baum ist eine zu verwaltende Ressource, die die unterschiedlichsten Typen von Ressourcen widerspiegelt. Ein Objekt kann sowohl ein Container sein, in dem weitere Objekte verwaltet werden, als auch ein Benutzer oder eine Gruppe sein. Eines ist bei allen Objekten aber immer gleich: Alle Objekte haben Eigenschaften, die *Attribute*. Die Attribute unterscheiden sich je nachdem, um welche Art von Objekt es sich handelt. Das Attribut, an dem ein Objekt erkannt wird, ist der *Common Name* (CN) eines Objekts. Über dieses Attribut wird das Objekt im DIT verwaltet.

Ein Objekt kann für die Verwaltung der unterschiedlichsten Aufgaben verwendet werden. Für jede Aufgabe benötigt ein Objekt unterschiedliche Attribute, deshalb werden Attribute zu Objektklassen zusammengefasst. Wenn zum Beispiel für einen Benutzer ein Objekt

erstellt werden soll, das für die Anmeldung an einem UNIX-System benötigt wird, muss dem Objekt die Objektklasse `posixAccount` zugeordnet werden, da diese die Attribute für eine erfolgreiche Anmeldung enthält. Listing 17.1 zeigt ein Beispiel für eine `objectclass`. Hier wird auch deutlich, dass bestimmte Attribute vergeben werden *müssen*, während andere vergeben werden *können*:

```
objectclass ( 1.3.6.1.1.1.2.0 NAME 'posixAccount' SUP top AUXILIARY
             DESC 'Abstraction of an account with POSIX attributes'
             MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
             MAY ( userPassword $ loginShell $ gecos $ description ) )
```

Listing 17.1 Beispiel für eine Objektklasse

Alle Attribute, die hier hinter `MUST` stehen, müssen zwingend vergeben werden, wenn einem Objekt die Objektklasse `posixAccount` zugeordnet wird. Die Attribute, die in der Zeile `MAY` stehen, können zusätzlich vergeben werden.

17.1.5 Attribute

Attribute sind die Eigenschaften von Objekten. Ein Attribut besteht aus einem Namen, mit dem das Attribut innerhalb eines Objekts eindeutig referenziert werden kann. Attribute können eine unterschiedliche Anzahl von Werten besitzen. Es gibt Attribute, die nur genau einen Wert haben dürfen, und es gibt Attribute, die mehrere Werte haben können. Um die Wiederverwendbarkeit von Attributen in verschiedenen Objektklassen zu ermöglichen, werden Attribute getrennt von Objekten verwaltet, und zwar in Form von Attributtypen. Der *attributetype* enthält die Komponenten, die Sie in Listing 17.2 sehen:

```
attributetype ( 1.3.6.1.1.1.1.4 NAME 'loginShell'
               DESC 'The path to the login shell'
               EQUALITY caseExactIA5Match
               SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

Listing 17.2 Beispiel für ein Attribut

In der ersten Zeile werden der *Object Identifier* (OID) und der Name des Attributs angegeben. Beide Werte müssen im gesamten DIT eindeutig sein. Eine Beschreibung aller OIDs können Sie unter www.alvestrand.no nachlesen.



Im Anschluss folgt eine Beschreibung, die frei formuliert werden kann. Danach gibt es die Möglichkeit, die Gleichheit (*EQUALITY*) für die Suche nach Attributen festzulegen. Ein Objekt mit dem Attribut `loginShell` würde nur angezeigt, wenn der Suchbegriff exakt, inklusive der Groß- und Kleinschreibung, übereinstimmt. Danach folgt noch eine Syntaxbeschreibung in Form eines OIDs. Alle möglichen OIDs können Sie im RFC 2252 nachlesen. Durch den OID wird festgelegt, um was für eine Art von Objekt es sich hier handelt, zum Beispiel um eine Zahl, eine Zeichenkette oder ein anderes Objekt.

17.1.6 Das Schema

Die Objektklassen werden dem LDAP-Server nicht einzeln zugeordnet, sondern in Gruppen zu einem Schema zusammengefasst. Für Schemata gilt das Gleiche wie für Objektklassen: Die Standardschemata sollten nicht erweitert werden, da es sonst bei einer eventuellen Zusammenführung zweier Bäume zu Konflikten kommt. Wenn Sie eigene Attribute benötigen, sollten Sie immer eine eigene Objektklasse in einem eigenen Schema erzeugen. Ein eigenes Schema können Sie auch gut in einen beliebigen LDAP-Baum integrieren.

Ein wichtiger Punkt bei der Erstellung eines eigenen Schemas ist der OID für die Attribute und Objektklassen. Natürlich können Sie den OID selbst wählen. Was tun Sie, wenn der selbst gewählte OID schon vergeben ist und später eine Zusammenführung gerade mit diesem Baum stattfinden soll? Deshalb sollten Sie Ihren OID immer registrieren. Den eigenen OID können Sie kostenlos unter der URL <http://pen.iana.org/pen/PenApplication.page> registrieren. Mit diesem Formular erhalten Sie einen OID vom Typ 1.3.6.1.4.1.*. Darunter können Sie nun die eigenen Attribute erstellen und nummerieren.

Neben einem eigenen OID sollten Sie für die Namen ein Präfix festlegen, mit dem alle Attributnamen im eigenen Schema beginnen. Wollen Sie zum Beispiel ein Schema für die Verwaltung von Tauchgebieten erstellen, kann jedes Attribut mit dem Präfix »DivingPlace« beginnen. Beispiele für Attributnamen wären »NAME DivingPlaceName«. Listing 17.3 zeigt ein Beispiel für ein eigenes Schema:

```
# Schema for Adminbuch about Divesites
#
# Last change: November 15, 2022
# Created by: Stefan Kania <stefan@example.net>
#
# General guideline:
# 1. The language in this file is english
# 2. Every OID in this file must look like this: ns.a.b.c., where
#    ns - the official namespace of the DiveSite schema:
#        1.3.6.1.4.1.12345
#    a - Reserved, must always be 1 for the DiveSite schema.
#    b - Entry type (1:attribute, 2:object)
#    c - Serial number (increased with every new entry)
# 3. Every entry in this file MUST have a "DESC" field, containing a
#    suitable description!
# 4. Attributes are listed in front of objects. All entries must be
#    ordered by their serial number.
# 5. All attributenames must start with 'DivingPlace'
#
# This schema is not depending on other schemas
```



```
# Attribute type definitions
attributetype (1.3.6.1.4.1.12345.1.1.1
  NAME 'DivingPlaceName'
  DESC 'Name of the diveplace'
  EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{100}
  SINGLE-VALUE)

attributetype (1.3.6.1.4.1.12345.1.1.2
  NAME 'DivingPlacePhoto'
  DESC 'JPEG photo of the diveplace'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.28
  SINGLE-VALUE)

attributetype (1.3.6.1.4.1.12345.1.1.3
  NAME 'DivingPlaceStreetName'
  DESC 'streetname of the diveplace'
  EQUALITY caseIgnoreListMatch
  SUBSTR caseIgnoreListSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.41
  SINGLE-VALUE)

attributetype (1.3.6.1.4.1.12345.1.1.4
  NAME 'DivingPlaceZipCode'
  DESC 'Zipcode of diveplace'
  EQUALITY caseIgnoreListMatch
  SUBSTR caseIgnoreListSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.41
  SINGLE-VALUE)

attributetype (1.3.6.1.4.1.12345.1.1.5
  NAME 'DivingPlaceCity'
  DESC 'Cityname of the diveplace'
  EQUALITY caseIgnoreListMatch
  SUBSTR caseIgnoreListSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.41
  SINGLE-VALUE)

attributetype (1.3.6.1.4.1.12345.1.1.6
  NAME 'DivingPlaceCountry'
  DESC 'Countryname of the DivingPlace'
  EQUALITY caseIgnoreListMatch
  SUBSTR caseIgnoreListSubstringsMatch
```

```
SYNTAX 1.3.6.1.4.1.1466.115.121.1.41
SINGLE-VALUE)

attributetype (1.3.6.1.4.1.12345.1.1.7
  NAME 'DivingPlacePhonenumber'
  DESC 'Official phonenumber of the diveplace'
  EQUALITY telephoneNumberMatch
  SUBSTR telephoneNumberSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.50)

attributetype (1.3.6.1.4.1.12345.1.1.8
  NAME 'DivingPlaceHomepage'
  DESC 'Official homepage of the diveplace'
  EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{255})

attributetype (1.3.6.1.4.1.12345.1.1.9
  NAME 'DivingPlaceMail'
  DESC 'Mailaddress of the diveplace'
  EQUALITY caseIgnoreMatch
  SUBSTR caseIgnoreSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

attributetype ( 1.3.6.1.4.1.12345.1.1.10
  NAME 'DivingPlaceOK'
  DESC 'If set to true then the diveplace is ok'
  EQUALITY booleanMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
  SINGLE-VALUE )

# Objectclass definitions
objectclass ( 1.3.6.1.4.1.12345.1.2.1
  NAME 'DiveSite'
  DESC 'Objectclass to manage all divesites'
  SUP top STRUCTURAL
  MUST (DivingPlaceName $ DivingPlaceCity)
  MAY (DivingPlacePhoto $ DivingPlaceStreetName $ DivingPlaceZipCode
    $ DivingPlaceCity $ DivingPlaceCountry $ DivingPlacePhonenumber
    $ DivingPlaceHomepage $ DivingPlaceMail $ DivingPlaceOK))
```

Listing 17.3 Beispiel für ein eigenes Schema



Achten Sie darauf, dass die Zeilen nach `attributetype` und `objectclass` eingerückt sind, sonst kommt es beim Start des LDAP-Servers zu Fehlern. Jedes der hier verwendeten Attribute kann sowohl über den OID als auch über den Namen eindeutig referenziert werden. Durch die Verwendung eines eigenen Suffixes kann es auch nicht zu Verwechslungen mit Standardattributen kommen. Durch die Verwendung von `SUP top STRUCTURAL` als Typ ist die Objektklasse von keiner anderen Objektklasse abhängig, und somit kann ein Objekt erstellt werden, das nur aus dieser Objektklasse besteht. Die komplette Beschreibung, wie Sie ein eigenes Schema aufbauen können, inklusive aller Syntaxbeschreibungen, finden Sie im RFC 2252.

17.1.7 Das LDIF-Format

Damit Sie nach der Konfiguration des LDAP-Servers die ersten Objekte in dem Baum erstellen können, benötigen Sie Dateien im *Lightweight Database Interchange Format* (LDIF).

In diesen Dateien werden die Objektklassen und Attribute mit ihren Werten für das zu erstellende Objekt angelegt und anschließend mit dem Kommando `ldapadd` in den Verzeichnisbaum eingetragen. Listing 17.4 zeigt zwei Beispiele für LDIF-Einträge:

```
dn: ou=users,dc=example,dc=net
ou: users
objectClass: organizationalUnit

dn: uid=skania,ou=users,dc=example,dc=net
uid: skania
cn: Stefan
sn: Kania
userPassword: {crypt}wUnJ/MvLSCoPQ
loginShell: /bin/bash
uidNumber: 501
gidNumber: 100
homeDirectory: /home/skania
shadowMin: -1
shadowMax: 999999
shadowWarning: 7
shadowInactive: -1
shadowExpire: -1
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
```

Listing 17.4 Beispiele für eine LDIF-Datei

Im ersten Teil wird eine neue organisatorische Einheit (OU) erzeugt, in der später die Benutzer erstellt werden sollen. Als erste Zeile muss immer der DN des Objekts stehen. Das Objekt

gehört zu den Objektklassen `top` (wobei Sie die Objektklassen nicht angeben müssen, sie wird automatisch hinzugefügt) und `organizationalUnit`. Nach dem letzten Attribut folgt eine Leerzeile. Diese Leerzeile dient als Trennung zwischen zwei Objekten und ist immer dann erforderlich, wenn mit einer LDIF-Datei mehrere Objekte erzeugt werden sollen.

Bei dem zweiten Objekt handelt es sich um einen Benutzer. Hier sehen Sie, wie das Objekt aus mehreren Objektklassen zusammengesetzt wird. Sie sehen aber auch, dass nicht alle Attribute aus den Objektklassen Verwendung finden. Mithilfe der LDIF-Dateien können Sie beliebig viele Objekte auf einmal erstellen oder verändern. Zusammen mit einem Shell-Skript wäre es auch möglich, die entsprechenden Werte für die Attribute aus einer anderen Datei auszulesen und damit komplexe Änderungen an vielen Objekten innerhalb des gesamten DIT durchzuführen.



Beachten Sie folgenden Hinweis

Wenn Sie im Verlauf dieses Kapitels LDIF-Dateien erstellen, achten Sie darauf, dass Sie am Ende einer Zeile keine Leerzeichen haben. Leerzeichen am Zeilenende sind Sonderzeichen, und Attribute mit Sonderzeichen werden im LDAP BASE64-kodiert abgelegt. Wenn Sie mit dem vi arbeiten, können Sie sich das Zeilenende mit der Option `:set list` anzeigen lassen.

Sollten Sie, gerade bei ACLs, längere Zeilen haben, und möchten Sie die Zeile gern umbrechen, ist das möglich. Nach dem Umbruch muss die nächste Zeile unbedingt mit einem Leerzeichen beginnen. Daran wird der Zusammenhang zur vorherigen Zeile erkannt.

17.2 Zu den hier verwendeten Distributionen

Bei der OpenLDAP-Installation verwenden wir nur noch Debian und Ubuntu, denn CentOS unterstützt den OpenLDAP-Server nicht mehr. Auch SUSE stellt bei SLES den OpenLDAP-Server nicht mehr bereit. Damit ist davon auszugehen, dass auch in der openSUSE-Version über kurz oder lang die Pakete entfernt werden. Beide Hersteller setzen auf den *389 DS*, der aus dem alten *Netscape Directory Server* entstanden ist.

Der *Netscape Directory Server* wurde von AOL gekauft und weiterentwickelt, später dann an Red Hat verkauft. Red Hat hat dann daraus einen eigenen LDAP-Server gebaut. Bei SUSE und Red Hat bleibt Ihnen immer noch der Weg über die Symas-Pakete, denn für einige Versionen der Distribution werden auch dort Pakete bereitgestellt.

Im ersten Schritt soll der OpenLDAP-Server installiert und konfiguriert werden. Wir werden Ihnen zeigen, wie Sie die Grundinstallation sowohl über die statische als auch über die dynamische Konfiguration einrichten können.

17.3 Installation der Symas-Pakete

Wenn Sie bereits einen OpenLDAP-Server auf einer der gängigen Distributionen installiert haben, jetzt aber die Symas-Pakete nutzen wollen, werden Sie in diesem Abschnitt alle Schritte für die Installation finden. Es gibt die folgenden Unterschiede zwischen den Distributionspaketen und den Symas-Paketen:

- ▶ Alle Dateien aus den Paketen werden im Verzeichnis `/opt/symas` abgelegt.
- ▶ Der Dienst starte nach der Installation als Benutzer `root`.
- ▶ Für die Grundkonfiguration müssen Sie selber sorgen.

In Listing 17.5 sehen Sie ein Skript zur Installation der Symas-Pakete:

```
#!/bin/bash
DEBIAN_FRONTEND=noninteractive apt install -y gnupg2 argon2 python3-ldap
wget -O- https://repo.symas.com/repo/gpg/RPM-GPG-KEY-symas-com-signing-key \
    | gpg --dearmor | tee /etc/apt/trusted.gpg.d/symas-com-gpg.gpg > /dev/null
echo "deb [arch=amd64] https://repo.symas.com/repo/deb/main/release26 \
    bullseye main" | tee -a /etc/apt/sources.list.d/symas26.list
apt update -y
groupadd -r -g 11 openldap
useradd -r -u 11 -g openldap -d /opt/symas/ openldap
DEBIAN_FRONTEND=noninteractive apt install -y symas-openldap-clients \
    symas-openldap-server
cp ./symas-openldap-server.service /lib/systemd/system/
systemctl daemon-reload
chown openldap:openldap /var/symas/openldap-data/
chmod 770 /var/symas/openldap-data/
chown openldap:openldap /var/symas/run
chmod 770 /var/symas/run
mkdir -m 770 /opt/symas/etc/openldap/slapd.d
chown openldap:openldap /opt/symas/etc/openldap/slapd.d
cp ./symas-openldap /etc/default
cp ./slapd.conf /opt/symas/etc/openldap/
chown root:openldap /opt/symas/etc/openldap/slapd.conf
chmod 750 /opt/symas/etc/openldap/slapd.conf
systemctl restart symas-openldap-server.service
```

Listing 17.5 Skript zur Installation der Symas-Pakete. Sie finden es beim Download-Material.

Beachten Sie folgende Schritte:

- ▶ Bevor die eigentlichen Pakete installiert werden können, ist es notwendig, dass Sie noch Abhängigkeiten auflösen und zusätzlich benötigte Pakete installieren.

Eines der Pakete ist das Paket *argon2*. Bei diesem Paket handelt es sich um die aktuell sicherste Passwortverschlüsselung, die wir hier auch im Buch nutzen wollen.

- ▶ Im nächsten Schritt wird dann der GPG-Key der Firma Symas im System installiert.
- ▶ Erst jetzt werden die Paketquellen zum System hinzugefügt und mit `apt update -y` zum System hinzugefügt.
- ▶ Vor der Installation der Pakete wird noch ein System-Benutzer und eine System-Gruppe angelegt. Achten Sie darauf, dass die entsprechenden IDs in Ihrem System noch frei sind. Wie schon einleitend erklärt, laufen die Dienste von Symas standardmäßig unter der Kennung `root`. Es ist aber immer besser, einen Dienst unter einer nicht privilegierten Benutzer- und Gruppenkennung laufen zu lassen.
- ▶ Erst jetzt werden die beiden benötigten Pakete installiert (*symas-openldap-server* und *symas-openldap-clients*).

Im Anschluss an die Installation der Pakete werden über das Skript noch weitere Systemanpassungen durchgeführt. Als Erstes wird eine neue Servicedatei für den `systemd` in das Verzeichnis `/etc/systemd/system` kopiert. Den Inhalt der Datei sehen Sie in Listing 17.6:

```
[Unit]
Description=Symas OpenLDAP Server Daemon
After=network-online.target
Documentation=man:slapd
Documentation=man:slapd-config
Documentation=man:slapd-mdb

[Service]
Type=notify
EnvironmentFile=/etc/default/symas-openldap
ExecStart=/opt/symas/lib/slapd -d 0 -h $SLAPD_URLS -u $SLAPD_USER \
-g $SLAPD_GROUP $STAT_DYN $SLAPD_CONF

[Install]
WantedBy=multi-user.target
```

Listing 17.6 Inhalt des `systemd`-Skripts

Die Variablen des Skripts werden über die Konfigurationsdatei `/etc/default/symas-openldap` gesetzt. Den Inhalt der Datei sehen Sie in Listing 17.7:

```
SLAPD_URLS="ldap:/// ldapi:/// ldaps:///"
SLAPD_OPTIONS=""
SLAPD_USER="openldap"
SLAPD_GROUP="openldap"
# Select static "-f" or dynamic "-F"
```

```

STAT_DYN="-f"
# To use static configuration
SLAPD_CONF="/opt/symas/etc/openldap/slapd.conf"
# To use dynamic configuration
#SLAPD_CONF="/opt/symas/etc/openldap/slapd.d"

```

Listing 17.7 Variablendeklaration für das systemd-Skript

Die hier aktiven Werte sorgen später für die statische Konfiguration über eine *slapd.conf*.

Keine statische Konfiguration mehr

Dieses Beispiel einer *slapd.conf* und die nachfolgend eingerichtete statische Startart des OpenLDAP werden wir nur noch hier zeigen, im weiteren Verlauf werden wir ausschließlich die dynamische Konfiguration nutzen. Die statische Konfiguration ist *deprecated*.

In den folgenden Schritten des Skripts wird dafür gesorgt, dass die Berechtigungen – sowohl für die statische als auch für die dynamische Konfiguration – korrekt gesetzt sind. Wenn sie die Einstellungen so übernehmen, wird Ihr OpenLDAP-Server mit der statischen Konfiguration aus Listing 17.8 gestartet:

```

# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include      /opt/symas/etc/openldap/schema/core.schema
include      /opt/symas/etc/openldap/schema/cosine.schema
include      /opt/symas/etc/openldap/schema/nis.schema
include      /opt/symas/etc/openldap/schema/inetorgperson.schema

#TLSCertificateFile /opt/symas/etc/openldap/example-net-cert.pem
#TLSCertificateKeyFile /opt/symas/etc/openldap/example-net-key.pem
#TLSCACertificateFile /opt/symas/etc/openldap/cacert.pem

pidfile      /var/symas/run/slapd.pid
argsfile     /var/symas/run/slapd.args

# Read slapd.conf(5) for possible values
loglevel     256

# Load dynamic backend modules:
modulepath   /opt/symas/lib/openldap
moduleload   back_mdb.la
moduleload   argon2.la

```

```
# The maximum number of entries that is returned for a search operation
sizelimit 500

# The tool-threads parameter sets the actual amount of cpu's that is used
# for indexing.
tool-threads 1
password-hash {ARGON2}
#####
# Specific Backend Directives for 'other':
# Backend specific directives apply to this backend until another
# 'backend' directive occurs
#backend      <other>
#database config
#####
# Specific Directives for database #1, of type hdb:
# Database specific directives apply to this database until another
# 'database' directive occurs
database      mdb
maxsize       1073741824
# The base of your directory in database #1
suffix        "dc=example,dc=net"

# rootdn directive for specifying a superuser on the database. This is needed
# for sync repl.
rootdn        "cn=admin,dc=example,dc=net"
#rootpw       "geheim"
rootpw        "{ARGON2}$argon2i$v=19$m=4096,t=3,p=1$c2FsdHNhbHRzYWx0$\
              qQCkx9nMeFlaG004DUmPDgrlUbgMMu09T1+vQCFuyzw"

# Where the database file are physically stored for database #1
directory     "/var/symas/openldap-data"

# Indexing options for database #1
index         objectClass eq

# Save the time that the entry gets modified, for database #1
lastmod       on

# The userPassword by default can be changed
# by the entry owning it if they are authenticated.
# Others should not be able to see it, except the
# admin entry below
# These access lines apply to database #1 only
```



```
access to attrs=userPassword,shadowLastChange
    by anonymous auth
    by self write
    by * none
```

```
access to *
    by * read
```

```
access to dn.base="" by * read
```

Listing 17.8 Statische Grundkonfiguration

Erzeugen eines Argon2 Passwort-Hash

Um ein Argon2 Passwort-Hash zu erzeugen verwenden Sie das Kommando `echo -n "geheim" | argon2 "12345678" -e`. Wobei `geheim` das zu verschlüsselnde Passwort ist und `12345678` der salt-Wert ist der für die Erstellung benötigt wird. Mehr zum Thema *Argon2* finden Sie unter <https://de.wikipedia.org/wiki/Argon2>.

17.3.1 Die zwei Konfigurationsarten

Für die Konfiguration des OpenLDAP-Servers gibt es zwei verschiedene Möglichkeiten. Eine ist die klassische Variante über die Datei `slapd.conf`. Hierbei handelt es sich um eine statische Konfiguration. Jedes Mal, wenn Sie etwas an der Konfiguration geändert haben, muss der Dienst neu gestartet werden. Dafür ist der Aufbau der Konfiguration sehr einfach. Jede Änderung tragen Sie in die Konfigurationsdatei ein und starten den `slapd` neu. Änderungen können direkt in der Datei dokumentiert werden. Wenn Sie aber mehrere LDAP-Server im Netz betreiben, müssen Sie die Konfigurationsdatei auf allen Systemen anpassen und überall den Dienst neu starten.

Die andere Variante ist die dynamische Konfiguration, bei der alle Parameter der Konfiguration im LDAP-Baum selbst abgelegt werden. Wenn Sie diese Art der Konfiguration verwenden, ist es nicht mehr notwendig, den Dienst bei einer Änderung der Konfiguration neu zu starten. Alle Änderungen befinden sich im LDAP-Baum und sind somit sofort gültig. Gerade wenn Sie eine Multi-Provider-Umgebung einrichten wollen, sollten Sie unbedingt die dynamische Konfiguration einsetzen. Da in einer Multi-Provider-Umgebung die Konfigurationen immer identisch sind, können Sie hier sehr gut die Replikation der Konfiguration nutzen. So brauchen Sie später Änderungen an der Konfiguration nur auf einem Provider durchführen.

Reicht Ihnen ein Provider in Ihrem Netz, können Sie (zumindest im Moment noch) die statische Konfiguration nutzen. Sie sollten aber wissen, dass diese Art der Konfiguration *deprecated* ist und irgendwann nicht mehr unterstützt wird. Wir werden hier im Buch ausschließlich die dynamische Konfiguration einsetzen.



Besser gleich dynamisch

Wenn Sie eine komplett neue LDAP-Infrastruktur aufbauen wollen, empfiehlt es sich heute, auf jeden Fall die dynamische Konfiguration zu wählen.

Umstellung auf die dynamische Konfiguration

Wenn Sie unsere Skripte für die Einrichtung der Symas-OpenLDAP-Pakete verwenden, werden Sie feststellen, dass wir dort noch die statische Konfiguration eingetragen haben. Um jetzt auf die dynamische Konfiguration umzustellen, ändern Sie die Datei `/etc/default/symas-openldap` wie in Listing 17.14:

```
SLAPD_URLS="ldap:/// ldapi:/// ldaps:///"
SLAPD_OPTIONS=""
SLAPD_USER="openldap"
SLAPD_GROUP="openldap"
# Select static "-f" or dynamic "-F"
STAT_DYN="-F"
# To use static configuration
#SLAPD_CONF="/opt/symas/etc/openldap/slapd.conf"
# To use dynamic configuration
SLAPD_CONF="/opt/symas/etc/openldap/slapd.d"
```

Listing 17.9 Umstellung auf die dynamische Konfiguration

17.3.2 Die Datenbank-Backends

Für OpenLDAP gibt es verschiedene Datenbank-Backends, die Sie einsetzen können. Da wäre die *Berkeley Database* (bdb) und die modifizierte Version der bdb, die *Hierarchical Database* (hdb). Beide Datenbanktypen sollten bei einer Neuinstallation nicht mehr gewählt werden, da sie als *deprecated* gekennzeichnet sind. Aktuell ist der Datenbanktyp `mdb`, wir werden hier nur noch die `mdb`-Datenbanken nutzen. Eigentlich lautet die genaue Abkürzung *lmdb* für *Lightning Memory-Mapped Database*. Dieser Datenbanktyp wurde extra für OpenLDAP entwickelt. Er ist erheblich performanter als alle anderen Typen. Mehr zum Thema finden Sie unter https://de.wikipedia.org/wiki/Lightning_Memory-Mapped_Database. Dort finden Sie weitere Links, in denen der Aufbau des Datenbanktyps genauer beschrieben wird.

Es ist nicht möglich, den Datenbanktyp im laufenden Betrieb zu ändern. Sie können aber eine LDIF-Sicherung Ihrer Datenbank erstellen, dann die alte Datenbank löschen und den Datenbanktyp in der Konfiguration anpassen. Anschließend starten Sie den LDAP-Server neu und spielen die LDIF-Datei zurück. Der Datenbanktyp ändert nichts an der Struktur der Daten.

Anpassen der Datei »/etc/ldap/ldap.conf«

Damit Sie in Zukunft nicht immer den Kontext des LDAP-Baums und die URL des LDAP-Servers bei den Kommandos angeben müssen, ist es sinnvoll, jetzt schon diese Werte in die Datei `/opt/symas/etc/openldap/ldap.conf` einzutragen. Passend zu dem Beispiel hier im Buch sehen die beiden Zeilen wie folgt aus:

```
BASE dc=example,dc=net
URL ldap://<hostname>.example.net
```

Listing 17.10 Einträge in der »ldap.conf«

17.3.3 Grundkonfiguration des LDAP-Servers (statisch)

Jetzt wird die `slapd.conf` für den ersten Start des OpenLDAP-Servers vorbereitet. Ganz am Anfang der Datei befinden sich die Include-Einträge für die verschiedenen Schemata. Da einige der Schemata von anderen Schemata abhängig sind, ist hier die Reihenfolge sehr wichtig. Die Abhängigkeiten können Sie in den einzelnen Schemadateien nachlesen. Auch die eventuell selbst erstellten Schemata können Sie sofort mit einbinden. Alle Include-Einträge sehen so aus wie in Listing 17.11:

```
include      /opt/symas/etc/openldap/schema/core.schema
include      /opt/symas/etc/openldap/schema/cosine.schema
include      /opt/symas/etc/openldap/schema/nis.schema
include      /opt/symas/etc/openldap/schema/inetorgperson.schema
```

Listing 17.11 Include-Einträge der »slapd.conf«

Im nächsten Schritt legen Sie das Suffix des DIT und den Verwalter des DIT mit seinem Passwort in der `slapd.conf` fest. Da der OpenLDAP-Server mehrere Datenbanken verwalten kann, werden diese Werte immer in dem Bereich der Datenbank eingetragen.

Listing 17.12 enthält die Einträge für den Beispiel-DIT:

```
suffix      "dc=example,dc=net"
rootdn      "cn=admin,dc=example,dc=net"
#rootpw     "geheim"
rootpw      "{ARGON2}$argon2i$v=19$m=4096,t=3,p=1$...."
```

Listing 17.12 Suffix und Manager des DIT

Damit Sie sicher sein können, dass keine Objekte mehr in der Datenbank vorhanden sind, können Sie die aktuelle Datenbank, nachdem Sie den LDAP-Server mit `systemctl stop slapd` angehalten haben, mit dem Kommando `rm /var/symas/openldap-data/*` löschen. Anschließend wird der LDAP-Server wieder mit `systemctl start slapd` gestartet. So können Sie sicher sein, dass die Datenbank wirklich leer ist.

17.3.4 Grundkonfiguration des LDAP-Servers (dynamisch)

Wie Sie den OpenLDAP-Server von Grund auf mit der dynamischen Konfiguration einrichten können, werden wir Ihnen in diesem Abschnitt zeigen. Dazu verwenden wir die *LDIF*-Datei aus Listing 17.13, die Sie auch in den Downloadmaterialien zum Buch finden:

```
dn: cn=config
objectClass: olcGlobal
cn: config
olcLogLevel: sync
olcLogLevel: stats
olcPidFile: /var/symas/run/slapd.pid
olcArgsFile: /var/symas/run/slapd.args
olcToolThreads: 1

dn: cn=schema,cn=config
objectClass: olcSchemaConfig
cn: schema

dn: cn=module{0},cn=config
objectClass: olcModuleList
cn: module0
olcModulePath: /opt/symas/lib/openldap
olcModuleLoad: back_mdb
olcModuleLoad: back_monitor
olcModuleLoad: autoca.la
olcModuleLoad: otp.la
olcModuleLoad: argon2.la
olcModuleLoad: syncprov
olcModuleLoad: back_monitor
olcModuleLoad: accesslog.la

include: file:///opt/symas/etc/openldap/schema/core.ldif
include: file:///opt/symas/etc/openldap/schema/cosine.ldif
include: file:///opt/symas/etc/openldap/schema/nis.ldif
include: file:///opt/symas/etc/openldap/schema/inetorgperson.ldif
include: file:///opt/symas/etc/openldap/schema/dyngroup.ldif

dn: olcDatabase={-1}frontend,cn=config
objectClass: olcDatabaseConfig
objectClass: olcFrontendConfig
olcDatabase: {-1}frontend
olcSizeLimit: 500
olcAccess: {0}to *
```

```

    by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
    by * break
olcAccess: {1}to dn="" by * read
olcAccess: {2}to dn.base="cn=subschema" by * read
olcPasswordHash: {ARGON2}

dn: olcDatabase={0}config,cn=config
objectClass: olcDatabaseConfig
olcDatabase: {0}config
olcRootDN: cn=admin,cn=config
olcRootPW: {ARGON2}$argon2i$v=19$m=4096,t=3,p=1$cXdlcnJ0enV6dWlvMTIz\
           $G/10lynf7ygdz0tG+E7S1fBibsFs/L80AUSisiGl/v4
olcAccess: {0}to *
    by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
    by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net write
    by dn.exact=uid=repl-user,ou=users,dc=example,dc=net read

dn: olcDatabase={1}monitor,cn=config
objectClass: olcDatabaseConfig
olcDatabase: {1}monitor
olcAccess: {0}to dn.subtree="cn=monitor"
    by dn.exact=cn=admin,cn=config read
    by dn.exact=cn=admin,dc=example,dc=net read
    by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net read

dn: olcDatabase={2}mdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcMdbConfig
olcDatabase: {2}mdb
olcSuffix: dc=example,dc=net
olcRootDN: cn=admin,dc=example,dc=net
olcRootPW: {ARGON2}$argon2i$v=19$m=4096,t=3,p=1$cXdlcnJ0enV6dWlvMTIz\
           $G/10lynf7ygdz0tG+E7S1fBibsFs/L80AUSisiGl/v4
olcDbCheckpoint: 512 30
olcDbDirectory: /var/symas/openldap-data
olcDbIndex: default eq
olcDbIndex: objectClass
olcDbIndex: entryUUID
olcDbIndex: entryCSN
olcDbIndex: cn pres,eq,sub
olcDbIndex: uid pres,eq,sub
olcDbIndex: mail pres,eq,sub
olcDbIndex: sn pres,eq,sub

```

```
olcDbIndex: description pres,eq,sub
olcDbIndex: title pres,eq,sub
olcDbIndex: givenName pres,eq,sub
olcDbMaxSize: 85899345920
olcAccess: {0} to *
  by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
  by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net write
  by dn.exact=uid=repl-user,ou=users,dc=example,dc=net read
  by * break
olcAccess: {1}to dn.exact="dc=example,dc=net" by * read
olcAccess: {2} to attrs=userPassword
  by anonymous auth by self write by * none
olcLimits: {0} dn.exact="uid=repl-user,ou=users,dc=example,dc=net"
  time=unlimited
  size=unlimited
olcLimits: {1} dn.exact="uid=ldap-admin,ou=users,dc=example,dc=net"
  time=unlimited
  size=unlimited
```

Listing 17.13 Basis der dynamischen Konfiguration

Wenn Sie sich die Einträge ansehen, werden Sie viele der Ihnen bekannten Parameter aus der *slapd.conf* wiederfinden, nur dass Ihnen hier immer die Abkürzung *olc* (OpenLDAP Configuration) vorangestellt ist. Um Ihnen den Einstieg etwas zu erleichtern, folgt hier eine kurze Erklärung zu den einzelnen Bereichen.

Die Konfiguration ist in verschiedene Bereiche unterteilt, die durch ein *dn*: eingeleitet werden. Bei den Bereichen kann es sich um die Konfiguration einer Datenbank handeln, oder um die Konfiguration eines Overlays (mithilfe von Overlays können Sie zusätzliche Funktionen im OpenLDAP einrichten und konfigurieren). Immer wenn Sie für den entsprechenden Bereich eine Änderung durchführen wollen, erstellen Sie eine LDIF-Datei, die genau diesen Bereich anspricht. In der aktuellen Konfiguration finden Sie die folgenden Bereiche:

- ▶ *dn*: *cn=config*
Zusammen mit dem *dn*: *olcDatabase={-1}frontend,cn=config* spiegelt dieser Abschnitt den globalen Bereich der *slapd.conf* wider. In diesem Bereich können Sie das *Loglevel* festlegen und die Pfade zu den *pid*- und *args*-Files festlegen. Alle Einstellungen im globalen Bereich gelten immer für alle Datenbanken, die Sie später noch einrichten.
- ▶ *dn*: *cn=schema,cn=config*
Hier wird der Name für das Backend der Schemata definiert. Da im Gegensatz zur statischen Konfiguration die Schemainformationen nicht einfach in den Speicher geladen werden, sondern in ein eigenes Backend, können Sie so später Schemaänderungen durchführen, ohne dass der *slapd* neu gestartet werden muss. Wenn

Sie sich nach der Einrichtung des OpenLDAP-Servers mal den Inhalt des Verzeichnis `/opt/symas/etc/openldap/slapd.d` ansehen, werden Sie dort auch ein passendes Unterverzeichnis finden, in dem für jedes Schema, das Sie über eine der `include`-Zeilen einfügen (zum Beispiel `include: file:///opt/symas/etc/openldap/schema/core.ldif` eine eigene LDIF-Datei erzeugt wird.

- ▶ `dn: cn=module{0},cn=config`
Immer wenn Sie ein zusätzliches Modul für den OpenLDAP benötigen, tragen Sie dieses Modul in diesem Bereich ein.
- ▶ `dn: olcDatabase={-1}frontend,cn=config`
Zusammen mit `dn: cn=config` bildet dieser Bereich die globale Konfiguration des OpenLDAP. Wenn Sie die dynamische Konfiguration bereits im OpenLDAP 2.4 genutzt haben, kommt hier eine wichtige Änderung auf Sie zu: Wenn Sie einen speziellen Passwort-Hash nutzen wollen, müssen Sie diesen hier eintragen. Bei OpenLDAP 2.4 wurde der Hash noch im Bereich `dn: cn=config` definiert. Das führt beim OpenLDAP ab 2.5 zu einer Fehlermeldung.
- ▶ `dn: olcDatabase={0}config,cn=config`
Hierbei handelt es sich um die Datenbank für die Konfiguration. Die gesamte Konfiguration aus allen Bereichen wird dort abgelegt. Die Datenbank muss immer die Nummer 0 haben, da die Konfiguration immer die erste Datenbank ist, die bei Start des OpenLDAP geladen wird.
- ▶ `dn: olcDatabase={1}monitor,cn=config`
Hier finden Sie alle wichtigen Informationen über die Nutzung des OpenLDAP. Die Werte dieser Datenbanken können Sie in Ihrem Monitoring auswerten und so die Auslastung Ihres OpenLDAP-Servers überwachen.
- ▶ `dn: olcDatabase={2}mdb,cn=config`
Bei dieser Datenbank handelt es sich um die eigentliche Objektdatenbank: In dieser Datenbank werden Sie später alle Objekte anlegen. In diesem Bereich tragen Sie auch alle *Access Control Lists (ACL)* für die Zugriffsberechtigungen auf die Objekte an. Auch eine etwaige Replikation auf andere OpenLDAP-Server wird hier konfiguriert.

Achten Sie auf die Nummern der Datenbanken

Nur die Datenbank `dn: olcDatabase={0}config,cn=config` hat eine feste Nummer, und zwar immer die 0. Alle folgenden Datenbanken können bei Ihnen in der Konfiguration andere Nummern besitzen. Prüfen Sie also vor einer Änderung immer, ob Sie auch die korrekte Nummer angegeben haben. So hätten wir zum Beispiel auch erst die Objektdatenbank und dann die Monitoringdatenbank definieren können. Dann hätte die Objektdatenbank die Nummer 1 und die Monitoringdatenbank die Nummer 2.



In der Konfiguration befinden sich schon zwei ACLs für den Zugriff auf die Objektdatenbank. Die Erste ist die Regel für den `uid=repl-user,ou=users,dc=example,dc=net` mit der Berechtigung, alles lesen zu können. Dieser Benutzer wird später für die Replikation benötigt.

Zum anderen sehen Sie dort auch die Berechtigung für den Benutzer `uid=ldap-admin,ou=users,dc=example,dc=net`. Dieser Benutzer hat später an allen Objekten das Schreibrecht. Mit diesem Benutzer werden später alle Änderungen am LDAP vorgenommen.

Aber warum wird für den schreibenden Zugriff ein anderer Benutzer benötigt, der `rootdn` hat doch immer alle Rechte und kann auch nicht in seinen Rechten beschnitten werden? Genau das ist der Grund. Denn falls es zu einer Kompromittierung des Benutzers mit allen Rechten kommen sollte, ist es einfacher, einen Benutzer im LDAP zu ersetzen, als den `rootdn` anpassen zu müssen. Der `rootdn` sollte nie für Änderungen am LDAP verwendet werden – außer in Notfällen, zum Beispiel, wenn Sie durch eine ACL die Zugriffsrechte zu stark beschnitten haben. Mehr zum Thema ACLs erfahren Sie in Abschnitt 17.8, »Absichern des LDAP-Baums mit ACLs«.

Wenn Sie also die dynamische Konfiguration für den LDAP-Server nutzen, werden alle Informationen in einer eigenen Datenbank im LDAP abgelegt. Alle Einträge und Änderungen werden dann über LDIF-Dateien vorgenommen. Die Möglichkeit, Kommentare zur Konfiguration abzulegen, gibt es hier nicht.

Der große Vorteil der dynamischen Konfiguration ist der, dass der LDAP-Server nach einer Änderung keinen Neustart benötigt: Die Änderung ist sofort, nachdem Sie die LDIF-Datei eingespielt haben, wirksam. Die ACLs werden auch in der Datenbank verwaltet; sie werden daher auch über LDIF-Dateien angepasst. Wenn Sie mit grafischen Werkzeugen arbeiten wollen, dann achten Sie darauf, ob das Werkzeug die dynamische Konfiguration bearbeiten kann. Wenn Sie mehrere LDAP-Server einsetzen und diese vielleicht auch noch an unterschiedlichen Standorten stehen und Sie häufig Änderungen an der Konfiguration oder den ACLs vornehmen, dann ist die dynamische Konfiguration auf jeden Fall der bessere Weg. Hier können Sie dann auch die verschiedensten Automatisierungswerkzeuge (wie Ansible oder Puppet) einsetzen.

Einspielen der Konfiguration

Bevor Sie die Konfiguration einspielen, stellen Sie sicher, dass Sie in der Datei `/etc/default/symas-openldap` die dynamische Konfiguration aktiviert haben, so wie Sie es in Listing 17.14 sehen:

```
SLAPD_URLS="ldap:/// ldapi:/// ldaps://"  
SLAPD_OPTIONS=""  
SLAPD_USER="openldap"  
SLAPD_GROUP="openldap"  
# Select static "-f" or dynamic "-F"
```



```

STAT_DYN="-F"
# To use static configuration
#SLAPD_CONF="/opt/symas/etc/openldap/slapd.conf"
# To use dynamic configuration
SLAPD_CONF="/opt/symas/etc/openldap/slapd.d"

```

Listing 17.14 Umstellung auf die dynamische Konfiguration

Nachdem Sie die Pakete installiert haben, existiert noch keine Konfiguration für den OpenLDAP. Ein Versuch, den Dienst zu starten, wird daher scheitern. Vor dem Start ist es notwendig, dass Sie die Konfiguration erst in den LDAP einspielen. Jetzt ist das Problem nur, dass die Konfiguration im LDAP liegt, Einträge im LDAP aber nur mit `ldapadd` hinzugefügt werden können, wenn der `slapd` auch läuft. Der `slapd` kann aber nicht gestartet werden, weil keine Konfiguration vorhanden ist. Um aus diesem Dilemma herauszukommen, gibt es das Kommando `slapadd`. Mithilfe von `slapadd` können Sie direkt in die verschiedenen Datenbankdateien schreiben, ohne dass der `slapd` läuft. So können Sie jetzt die Grundkonfiguration wie in Listing 17.15 in den LDAP einspielen:

```

root@ldap01:~# slapadd -n0 -F /opt/symas/etc/openldap/slapd.d -l config.ldif
Closing DB...
root@ldap01:~# chown -R openldap: /opt/symas/etc/openldap/slapd.d
root@ldap01:~# systemctl restart symas-openldap-server.service

```

Listing 17.15 Einspielen der Grundkonfiguration

Um sich die Konfiguration ansehen zu können, können Sie am einfachsten über die lokale `ldapi:///`-Schnittstelle zusammen mit der externen Authentifizierung auf die Konfiguration zugreifen. Diese Art der Authentifizierung funktioniert nur direkt auf dem Server und nur als Benutzer `root`. In Listing 17.16 sehen Sie eine stark gekürzte Ausgabe der Konfiguration:

```

root@ldap01:~# ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config | less
# extended LDIF
#
# LDAPv3
# base <cn=config> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#
# config
dn: cn=config
objectClass: olcGlobal
cn: config
olcArgsFile: /var/symas/run/slapd.args
olcLogLevel: sync

```

```
olcLogLevel: stats
olcPidFile: /var/symas/run/slapd.pid
olcToolThreads: 1

# module0, config
dn: cn=module0,cn=config
objectClass: olcModuleList
cn: module0
olcModulePath: /opt/symas/lib/openldap
olcModuleLoad: 0back_mdb
olcModuleLoad: 1back_monitor
olcModuleLoad: 2autoca.la
olcModuleLoad: 3otp.la
olcModuleLoad: 4argon2.la
olcModuleLoad: 5syncprov
olcModuleLoad: 6back_monitor
olcModuleLoad: 7accesslog.la
...
# search result
search: 2
result: 0 Success

# numResponses: 13
# numEntries: 12
```

Listing 17.16 Ausgabe der Konfiguration

Das Kommando lässt sich nur als root ausführen. Die Berechtigung erhalten Sie über die Zeile aus Listing 17.17

```
olcAccess: {0}to * by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,
cn=auth manage by * break
```

Listing 17.17 ACL für den Admin-Zugriff

Hier die Erklärung zu dem Kommando:

- ▶ -Y EXTERNAL
Immer wenn der SASL-Mechanismus EXTERNAL Verwendung findet, wird für den angemeldeten Benutzer, der das Kommando ausführt, ein DN mit dem Wert `gidNumber=<GID>+uidNumber=<UID>,cn=peercred,cn=external,cn=auth` generiert.
- ▶ -H ldapi:///.
Diese Art der Authentifizierung kann nur über den lokalen Socket `ldapi:///` verwendet werden. Es wird immer die GID und UID des Benutzers verwendet, der das Kommando

ausführt. Da hier in der ACL die GID und die UID 0 angegeben ist, kann nur der Benutzer *root* diese Authentifizierung erfolgreich durchführen und so Zugriff auf die entsprechenden Datenbank erhalten.

► -b "cn=config"

Bei dem Backend *cn=config* handelt es sich um die dynamische Konfiguration, auf die zugegriffen wird.

Wenn Sie die Konfiguration einmal mit der Datei *slapd.conf* vergleichen, werden Sie dort alle Parameter wiederfinden. Einige Parameter kommen hier für die Konfiguration noch dazu. Damit haben Sie die Grundkonfiguration Ihres OpenLDAP-Servers abgeschlossen.

In der von uns vorbereiteten Konfiguration wurde das Passwort für den *rootdn* der Konfiguration und der Objektdatenbank auf *geheim* gesetzt. Das soll im nächsten Schritt geändert werden. Da sich die Informationen in einer Datenbank befinden, werden Sie diese und alle weiteren Änderungen an der Konfiguration immer über LDIF-Dateien eintragen. Eine LDIF-Datei für die Passwortänderung des *rootdn* sehen Sie in Listing 17.18:

```
dn: olcDatabase={0}config,cn=config
changetype: modify
replace: olcRootPW
olcRootPW: {ARGON2}$argon2i$v=19$m=4096,t=3,p=1...
```

Listing 17.18 LDIF-Datei für das Admin-Passwort

Das Passwort wurde im Übrigen zuvor mit dem Kommando `echo -n "supergeheim" | argon2 "saltsaltsalt" -e` generiert und in die LDIF-Datei eingetragen. Die Änderung tragen Sie am einfachsten über den SASL-Mechanismus *EXTERNAL* ein (siehe Listing 17.19):

```
root@ldap01:~# ldapmodify -Q -Y EXTERNAL -H ldapi:/// -f config-root-pw.ldif
modifying entry "olcDatabase={0}config,cn=config"
```

Listing 17.19 Eintragen des Adminpassworts

Nachdem Sie das Passwort für den *rootdn* der Konfigurationsdatenbank eingetragen haben, können Sie jetzt das Passwort für den *rootdn* der Objektdatenbank anpassen. In Listing 17.20 sehen Sie die LDIF-Datei für die Änderung:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
replace: olcRootPW
olcRootPW: {ARGON2}$argon2i$v=19$m=4096,t=3,p=1...
```

Listing 17.20 Änderung des Passworts in der Objektdatenbank

Die LDIF-Datei spielen Sie mit demselben Kommando ein, das Sie zuvor schon für die Änderung des Passworts in der Konfiguration verwendet haben. Wenn Sie beide LDIF-Dateien vergleichen, dann sehen Sie, dass der einzige Unterschied die zu ändernde Datenbank ist.

Modify-Typen

Bei den vorherigen LDIF-Dateien, die zur Änderung der Konfiguration genutzt wurden, sehen Sie verschiedene Modify-Typen. Die verschiedenen Typen haben folgende Funktion:

- ▶ `add`
Immer wenn Sie `add` nutzen, wird das entsprechende Attribut zu einem Objekt oder einer Konfigurationsdatenbank hinzugefügt. Ist das Attribut aber schon vorhanden, kommt es zu einer Fehlermeldung – es sei denn, es handelt sich um ein *multi-value*-Attribut, dann würde ein zusätzlicher Wert zu dem Attribut hinzugefügt.
- ▶ `replace`
Bei der Verwendung von `replace` wird ein bestehendes Attribut geändert, ist das Attribut nicht vorhanden, wird es hinzugefügt.
- ▶ `delete`
Irgendwann möchten Sie ein Attribut vielleicht auch mal entfernen. Dann ist der Modify-Typ `delete` der richtige Weg. Wollen Sie einen Wert eines *multi-value*-Attributs entfernen, ist es notwendig, dass Sie hier noch den Wert angeben den Sie entfernen wollen. Geben Sie kleinen Wert an, wird das gesamte Attribut mit allen Werten gelöscht.

17.3.5 Anlegen der ersten Objekte

Im nächsten Schritt sollen die ersten Objekte im DIT angelegt werden. Dazu erstellen Sie die LDIF-Datei wie in Listing 17.21:

```
# Entry 1: dc=example,dc=net
dn: dc=example,dc=net
objectClass: domain
objectClass: dcObject
dc: example

# Entry 2: ou=users,dc=example,dc=net
dn: ou=users,dc=example,dc=net
ou: users
objectClass: organizationalUnit

# Entry 3: ou=groups,dc=example,dc=net
dn: ou=groups,dc=example,dc=net
ou: groups
objectClass: organizationalUnit
```

Listing 17.21 LDIF-Datei für die Grundkonfiguration des DIT

Das erste Objekt bildet die oberste Ebene des DIT und ist ein Objekt vom Typ *domain component*. Bei den anderen zwei Objekten handelt es sich um OU-Objekte.

In diesen OUs sollen später die Benutzer und Gruppen verwaltet werden. Nachdem Sie die Datei erstellt haben, können Sie sie mit dem Kommando `ldapadd` einspielen. In Listing 17.22 sehen Sie das Beispiel für die Einspielung der Datei:

```
root@ldap01:~# ldapadd -Y EXTERNAL -H ldapi:/// -f home.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "dc=example,dc=net"

adding new entry "ou=users,dc=example,dc=net"

adding new entry "ou=groups,dc=example,dc=net"
```

Listing 17.22 Einspielen der Datei »home.ldif«

An dem Beispiel sehen Sie, dass Sie auch bei Änderungen an der Objektdatenbank die Authentifizierung über das SASL-Modul EXTERNAL nutzen können.

Weitere Änderungen

Jede weitere Änderung an der Konfiguration spielen Sie in Zukunft über eine LDIF-Datei ein. Im folgenden Beispiel sehen Sie, wie Sie das *LogLevel* anpassen können. Die Änderung soll hier in mehreren Schritten durchgeführt werden. Wie Sie in der Grundkonfiguration gesehen haben, haben wir dort bereits zwei LogLevel eingetragen. Jetzt soll das LogLevel sync entfernt werden. Als Erstes benötigen Sie eine LDIF-Datei in der Sie das LogLevel festlegen. In Listing 17.23 sehen Sie die benötigte Datei:

```
dn: cn=config
changetype: modify
delete: olcLogLevel
olcLogLevel: sync
```

Listing 17.23 Anpassen des Loglevels

Würden Sie in der LDIF-Datei nicht die Zeile `olcLogLevel: sync` verwenden, würden hier alle vorhandenen LogLevel aus der Konfiguration entfernt. Wie Sie die Änderungen einspielen, sehen Sie in Listing 17.24:

```
root@ldap01:~# ldapmodify -Y EXTERNAL -H ldapi:/// -f loglevel.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"
```

Listing 17.24 Einspielen des Loglevels

17.4 Die Verbindung zum LDAP-Server über TLS absichern

Bevor die ersten Benutzer angelegt und einen LDAP-Client für den Zugriff konfiguriert werden soll, wollen wir Ihnen jetzt zeigen, wie Sie eine verschlüsselte Verbindung zum LDAP-Server realisieren. Denken Sie immer daran: Ohne die Einrichtung und Nutzung einer Verschlüsselung, werden alle Daten, inklusive aller Passwörter, unverschlüsselt über das Netzwerk übertragen.

Das ist auch mit ein Grund, warum wir bis zu diesem Zeitpunkt alle Änderungen nur lokal auf dem LDAP-Server mit dem SASL-Mechanismus *EXTERNAL* durchgeführt haben, denn dabei werden keine Daten über das Netzwerk übertragen. Durch die Verwendung von TLS kann die Verbindung aber verschlüsselt werden.

TLS ist eine Weiterentwicklung des SSL-Protokolls. Das primäre Ziel von TLS besteht darin, die Integrität und eine geschützte Verbindung zwischen zwei kommunizierenden Anwendungen herzustellen. Hierzu wird eine Verschlüsselungsmethode verwendet, die auf Schlüsseln und Zertifikaten basiert. Das Zertifikat muss normalerweise von einer *Certificate Authority* (CA) beglaubigt werden. Da hier das Zertifikat aber nur intern verwendet wird, können Sie das Zertifikat selbst beglaubigen. Dieses Zertifikat würde außerhalb des Netzwerks als *self-signed* betrachtet und von gut konfigurierten Sicherheitssystemen abgelehnt werden. Aber die Verwendung außerhalb unseres Netzwerks ist auch nicht gewollt.

Zur Erstellung aller Zertifikate wird OpenSSL verwendet. OpenSSL wird über die Datei */etc/ssl/openssl.cnf* konfiguriert. In dieser Datei können Sie bestimmte Voreinstellungen vornehmen, die das Erstellen der Zertifikate erleichtern. Wichtig ist auf jeden Fall, dass vor der Zeile `# unique_subject = no` das Hash-Zeichen entfernt wird, da es sonst zu Fehlern beim Erstellen der Zertifikate kommt und die Zertifikate nach dem Ablauf nicht verlängert werden können. Mehr zum Thema Zertifikate finden Sie in Abschnitt 31.5.1, »Einrichtung einer PKI mit Server- und E-Mail-Zertifikaten«.

Bei Debian und Ubuntu befindet sich im Verzeichnis */usr/lib/ssl/misc/* das Programm *CA.pl*. Dieses Programm wird verwendet, um die eigenen Zertifikate zu erstellen. Da sich das Programm nicht im Pfad befindet, können Sie einfach einen Link in das Verzeichnis */usr/local/sbin* legen. Dann können Sie das Skript immer verwenden, ohne einen Pfad angeben zu müssen. In unserem Beispiel sollen alle Zertifikate im Verzeichnis */etc/ssl/zertifikate/* erzeugt werden. Legen Sie das Verzeichnis an, bevor Sie die ersten Zertifikate erzeugen. Anschließend wechseln Sie in das Verzeichnis, da das Programm *CA.pl* alle Zertifikate relativ zum aktuellen Standpunkt im Dateisystem erzeugt.

17.4.1 Erstellen der Zertifizierungsstelle

Als Erstes müssen Sie das Zertifikat für die Zertifizierungsstelle (CA) erzeugen. Das geht mit dem Kommando `CA.pl -newca`. Damit wird das Verzeichnis */etc/ssl/zertifikate/demoCA/* er-

stellt. In diesem Verzeichnis werden die benötigten Zertifikate abgelegt. Beim Erzeugen des Zertifikats werden die Parameter für das Zertifikat interaktiv abgefragt. Einige Grundeinstellungen, wie zum Beispiel zur Firma und zum Ort, können Sie über die Datei *openssl.cnf* bereits voreinstellen. Andere Parameter, wie beispielsweise der `common name` und die Mail-Adresse, müssen beim Erzeugen des Zertifikats angegeben werden. Für die *demoCA* ist es nicht relevant, welchen `common name` Sie vergeben.

17.4.2 Erstellen des Serverzertifikats

Im nächsten Schritt wird das Serverzertifikat erzeugt. Dabei ist es absolut wichtig, dass Sie hier als `common name` den *FQDN* des Servers angeben, da das Zertifikat sonst nicht zum Server passt und damit ungültig ist. Das Zertifikat erstellen Sie mit dem Kommando `CA.pl -newreq`.

Wildcard-Zertifikate nutzen

Wenn Sie im *FQDN* für den Hostnamen den `*` nutzen (z.B. `*.example.net`), können Sie später das Zertifikat auf allen LDAP-Servern nutzen und müssen nicht für jeden Server ein eigenes Zertifikat erstellen. Diese Option sollten Sie aber nur in Ihrer Testumgebung einsetzen, da dieses Zertifikat auch für andere Dienste verwendet werden könnte und somit zu einer Sicherheitslücke führen würde.



17.4.3 Signieren des Zertifikats

Im Anschluss daran signieren Sie das gerade erstellte Zertifikat mit dem Zertifikat der *demoCA*. Das geschieht mit dem Kommando `CA.pl -sign`. Beim Erstellen des Serverzertifikats haben Sie ein Passwort für den Schlüssel des Zertifikats eingegeben. Dieses Passwort muss wieder entfernt werden. Ansonsten wird beim Neustart des Servers der LDAP-Dienst nicht gestartet, da das Passwort nicht eingegeben wurde. Das Passwort entfernen Sie mit dem Kommando `openssl rsa -in newkey.pem -out ldapkey.pem`. Dadurch wird auch gleich der Zertifikatsschlüssel umbenannt, damit es später, wenn noch mehr Zertifikate in dem Verzeichnis abgelegt werden, nicht zu Verwechslungen kommt.

Im Verzeichnis */etc/ssl/zertifikate/* existiert jetzt noch eine Datei *newcert.pem*. Dabei handelt es sich um das Zertifikat für den LDAP-Server. Diese Datei sollten Sie nun in *ldapcert.pem* umbenennen, um auch hier Verwechslungen zu vermeiden.

Jetzt liegen sowohl das benötigte Zertifikat als auch der dazugehörige Zertifikatsschlüssel im selben Verzeichnis. Achten Sie zum Abschluss darauf, dass der Benutzer `openldap` sowohl das Zertifikat als auch den dazugehörigen Schlüssel lesen darf. Setzen Sie eventuell die Rechte, bevor Sie mit dem nächsten Schritt fortfahren. Das Zertifikat können Sie jetzt mit dem Kommando `CA.pl -verify` überprüfen. Damit ist die Erstellung des Zertifikats abgeschlossen.

17.4.4 Zertifikate in die »slapd.conf« eintragen

Wenn Sie die statische Konfiguration nutzen, besteht Ihr nächster Schritt darin, die Zeilen aus Listing 17.25 in den globalen Bereich der Datei *slapd.conf* einzutragen:

```
TLSCertificateFile /etc/ssl/zertifikate/ldapcert.pem
TLSCertificateKeyFile /etc/ssl/zertifikate/ldapkey.pem
TLSCACertificateFile /etc/ssl/zertifikate/demoCA/cacert.pem
TLSVerifyClient allow
```

Listing 17.25 Einträge für TLS in der »slapd.conf«

Im Anschluss daran starten Sie den LDAP-Dienst neu, damit die neuen Parameter gültig werden.

17.4.5 Zertifikate in die dynamische Konfiguration eintragen

Auch bei der Verwendung von TLS wollen wir auf die dynamische Konfiguration eingehen. An der Erstellung der Zertifikate ändert sich nichts, auch die spätere Anpassung für die Clientprogramme in der Datei */etc/ldap/ldap.conf* ist identisch mit der statischen Konfiguration. Nur das Verfahren, wie die Einträge in die Konfiguration gelangen, ist hier anders. Sie haben es sich bestimmt schon gedacht: Auch hier benötigen Sie eine LDIF-Datei, in der die Parameter eingetragen sind, die Sie dann mit *ldapmodify* in die Konfiguration übernehmen können. In Listing 17.26 sehen Sie die entsprechende LDIF-Datei:

```
dn: cn=config
changetype: modify
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/zertifikate/provider-dyn-cert.pem
-
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/zertifikate/provider-dyn-key.pem
-
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/zertifikate/demoCA/cacert.pem
```

Listing 17.26 LDIF-Datei für die TLS-Einträge

Spielen Sie die Datei genau so ein, wie Sie es schon mit der LDIF-Datei für das Loglevel gemacht haben. Schauen Sie sich anschließend die Konfiguration an, und prüfen Sie, ob und wo die Einträge durchgeführt wurden.

Nutzung von replace

Anstelle von `add` können Sie auch `replace` nutzen. Auch wenn noch kein Zertifikat in die Konfiguration eingetragen ist, würden die entsprechenden Pfade übernommen werden. Der Vorteil wäre, dass Sie die LDIF-Datei für spätere Änderungen an den Pfaden wiederverwenden könnten.

Jetzt spielen Sie die Änderungen mit dem Kommando `ldapmodify -Y EXTERNAL -H ldapi:/// -f tls.ldif` ein. Sollten Sie beim Versuch, die Änderungen in die Konfiguration einzutragen, die Fehlermeldung aus Listing 17.27 bekommen, dann prüfen Sie die Zugriffsrechte auf alle Zertifikatsdateien. Prüfen Sie auch, ob die Gruppe, unter der Ihr LDAP-Server läuft, Zugriff auf die Dateien für das Zertifikat besitzt. Prüfen Sie auch die Pfade.

```
modifying entry "cn=config"
ldap_modify: Other (e.g., implementation specific) error (80)
```

Listing 17.27 Fehler bei falschen Berechtigungen

Die Einträge für die TLS-Verschlüsselung finden Sie gleich am Anfang der dynamischen Konfiguration – also, wie bei der statischen Konfiguration, im globalen Bereich der Konfiguration. Das ist auch sinnvoll, da sich die Verschlüsselung auf alle Zugriffe auf den Server bezieht und nicht auf die Zugriffe auf einzelne Datenbanken.


17

17.4.6 Konfiguration des LDAP-Clients

Damit jetzt auch der LDAP-Client das Zertifikat prüfen kann, muss der Client das Zertifikat der Zertifizierungsstelle kennen und wissen, dass der Server TLS unterstützt. Dazu tragen Sie die folgenden zwei Zeilen aus Listing 17.28 in die Datei `ldap.conf` ein:

```
TLS_REQCERT demand
TLS_CACERT /etc/ssl/zertifikate/demoCA/cacert.pem
```

Listing 17.28 Änderungen an der Datei »ldap.conf«

Durch die Verwendung von `TLS_REQCERT demand` wird der Zugriff auf den Server abgelehnt, wenn das Zertifikat auf dem Server abgelaufen oder fehlerhaft ist. 

Im Anschluss daran können Sie die TLS-Konfiguration mit dem Kommando `ldapsearch -d 1 -x -ZZ` testen. Die Option `-ZZ` sorgt dafür, dass der Client die Verbindung zum Server nur akzeptiert, wenn der Server die Verschlüsselung unterstützt. Durch die Option `-d 1` wird der Debug-Modus aktiviert, und Sie können den TLS-Verbindungsaufbau überprüfen.



Verwendung von »ldaps«

Wollen Sie den Zugriff über das Protokoll *ldaps* zulassen, dann passen Sie die Datei */etc/default/slapd* wie folgt an:

```
SLAPD_SERVICES="ldap:/// ldapi:/// ldaps://"
```

Starten Sie anschließend den *slapd* neu. Wenn Sie nun in der Prozessliste nach dem *slapd* suchen, werden Sie sehen, dass jetzt zusätzlich das Protokoll *ldaps:///* mit angegeben ist.

Wenn Sie die Clientprogramme wie *ldapsearch* immer mit *ldaps* nutzen wollen, ändern Sie den Eintrag für die URI in der Datei */etc/ldap/ldap.conf* entsprechend. Den Parameter *-ZZ* dürfen Sie im Anschluss nicht mehr verwenden, das würde zu einer Fehlermeldung führen.

Von nun an können alle Clientanwendungen über eine gesicherte TLS-Verbindung auf den Server zugreifen. Prüfen Sie bei den grafischen Werkzeugen ob die Werkzeuge selbstständig erkennen, dass der Server TLS unterstützt, oder ob Sie das Werkzeug für den TLS-Zugriff konfigurieren müssen.



startTLS oder ldaps

Was ist denn nun der richtige Weg für die Verschlüsselung? Sollte *startTLS* zusammen mit dem Protokoll *ldap* über den Port 389 genutzt werden, oder doch besser das Protokoll *ldaps* über den Port 636? Bis vor einiger Zeit hieß es noch: »Nutzen Sie *startTLS*, das ist aktueller und sicherer.« Inzwischen wurde die Sicherheit von *ldaps* allerdings verbessert, und im Gegenzug wurden Sicherheitslücken bei der Verwendung von *startTLS* gefunden. Aus diesem Grund gilt heute die Empfehlung, wieder das Protokoll *ldaps* zu nutzen. Deswegen wir bei unseren Beispielen ausschließlich das Protokoll *ldaps* nutzen. Aber egal, welche Art der Verschlüsselung Sie wählen: Hauptsache ist, dass Sie Ihre Kommunikation mit dem LDAP-Server überhaupt verschlüsseln.

17.5 Einrichtung des *sssd*

Bis vor einigen Jahren wurde der LDAP-Client über die Pakete *libpam-ldap* und *libpam-nss* eingerichtet, aber seit der *sssd* in allen Distributionen Einzug gehalten hat, ist es auf jeden Fall sinnvoller, ihn als LDAP-Client zu verwenden. Der *sssd* hat gegenüber der alten Clientkonfiguration die folgenden Vorteile:

- ▶ Sie können mehrere Verzeichnisdienste einfach in einer Konfigurationsdatei ablegen und somit gleichzeitig unterschiedliche Authentifizierungsquellen nutzen.
- ▶ Der *sssd* unterstützt die TLS-Verschlüsselung (und die Verschlüsselung über das Protokoll *ldaps*), ohne weitere Konfigurationen vorzunehmen.

- ▶ Die Konfigurationsdatei ist vor fremdem Zugriff geschützt, da die Rechte der Datei auf 600 gesetzt sein müssen und die Datei dem Benutzer *root* gehören muss. Wenn diese Bedingungen nicht erfüllt sind, startet der Dienst nicht.
- ▶ Der *sssd* ist ein eigenständiger Daemon und unterstützt verschiedene Authentifizierungsquellen.
- ▶ Durch die umfangreichen Konfigurationsmöglichkeiten können Sie den *sssd* sehr gut an Ihre Umgebung anpassen.
- ▶ Die Datei *nsswitch.conf* wird automatisch bei der Installation angepasst.
- ▶ Der *sssd* unterstützt die Weiterleitung an Referrals.

Als Erstes installieren Sie die Pakete *sssd* und *sssd-tools*. Alle zusätzlich benötigten Pakete werden als Abhängigkeiten automatisch installiert.

Das Paket »ldap-utils«

Bei der Installation des *sssd* wird bei den Abhängigkeiten auch das Paket *ldap-utils* installiert. Dieses Paket steht aber im Konflikt zu den gleichnamigen Kommandos in den Symas-Paketen. Nachdem Sie den *sssd* installiert haben, ist es unbedingt notwendig, dass Sie das Paket *ldap-utils* deinstallieren.

17

Während der Installation der Pakete wird gleich die Datei */etc/nsswitch.conf* angepasst. Listing 17.29 zeigt Ihnen die durchgeführten Änderungen:

```
passwd:      compat systemd sss
group:       compat systemd sss
shadow:     compat sss
gshadow:    files

hosts:      files dns
networks:   files

protocols:  db files
services:  db files sss
ethers:     db files
rpc:        db files

netgroup:   nis sss
sudoers:    files sss
```

Listing 17.29 Die neue »nsswitch.conf«

Der Eintrag `sss` sorgt dafür, dass der `sssd` zusätzlich zu den lokalen Datenbanken verwendet wird. Da der `sssd` den LDAP-Baum nach Benutzern durchsuchen muss, benötigen Sie ein Benutzerobjekt, mit dem der `sssd` den Baum durchsucht. Das ist besonders wichtig, wenn Sie Ihren LDAP-Baum mit ACLs abgesichert haben. Der `sssd` muss die Benutzerinformationen aller Benutzer und Gruppen lesen können. Verwenden Sie hier auf gar keinen Fall einen POSIX-Account für den Benutzer, sondern lediglich ein `SimpleSecurityObject`: Dann kann das Objekt nur zur Authentifizierung verwendet werden und nicht für ein Login. Erzeugen Sie ein `SimpleSecurityObject` mit der LDIF-Datei `sssd-user.ldif`. Das Passwort für das Objekt in unserem Beispiel ist geheim. Listing 17.30 zeigt den Inhalt der LDIF-Datei:

```
dn: uid=sssd-user,ou=users,dc=example,dc=net
objectClass: account
objectClass: simpleSecurityObject
objectClass: top
uid: sssd-user
userPassword: ARGON2....
```

Listing 17.30 LDIF-Datei für den »sssd-user«

Das neue Objekt spielen Sie genau so ein, wie Sie es schon mit den ersten Objekten gemacht haben. Da wir in der Grundkonfiguration schon die anonymen Zugriffe auf den LDAP-Baum gesperrt haben, ist es an dieser Stelle notwendig, dass Sie eine ACL für den Zugriff ergänzen. In Listing 17.31 sehen Sie die LDIF-Datei, um die ACL anzupassen. Wir werden später noch genauer auf die Änderung von ACLs eingehen. Im Moment geht es nur darum, dem `sssd-user` die richtigen Rechte zu geben.

```
dn: olcDatabase=2mdb,cn=config
changetype: modify
delete: olcAccess
olcAccess: 0
-
add: olcAccess
olcAccess: 0 to *
  by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
  by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net write
  by dn.exact=uid=sssd-user,ou=users,dc=example,dc=net read
  by dn.exact=uid=repl-user,ou=users,dc=example,dc=net read
  by * break
```

Listing 17.31 ACLs für den `sssd-user`

In der LDIF-Datei sehen Sie schon eine Besonderheit bei der Änderung von ACLs. Eine bestehende ACL kann nicht einfach ergänzt werden, sondern muss erst gelöscht und dann mit den neuen Werten neu angelegt werden.

Nachdem Sie den Benutzer und die ACL angelegt haben, können Sie jetzt die Konfigurationsdatei `/etc/sss/sss.conf` für den `sss` so wie in Listing 17.32 erstellen:

```
[sss]
config_file_version = 2
services = nss, pam
domains = EXAMPLE

[nss]
reconnection_retries = 4

[pam]
reconnection_retries = 4
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5

[domain/EXAMPLE]
ldap_schema=rfc2307
ldap_uri = ldaps://ldap01.example.net:636
ldap_search_base=dc=example,dc=net
ldap_default_bind_dn=uid=sss-user,ou=users,dc=example,dc=net
ldap_default_authtok=geheim
id_provider=ldap
auth_provider=ldap
chpass_provider = ldap
ldap_chpass_uri = ldaps://provider-dyn.example.net:636
cache_credentials = True
ldap_tls_cacertdir = /etc/ssl/demoCA
ldap_tls_cacert = /etc/ssl/zertifikate/demoCA/cacert.pem
ldap_id_use_start_tls = True
```

Listing 17.32 Die Datei »sss.conf«

Die richtigen Rechte setzen

Die Datei muss die Berechtigung 600 haben und dem Benutzer `root` und der Gruppe `root` gehören. Wenn Sie die Rechte hier nicht richtig setzen, startet der Dienst nicht.

Starten Sie jetzt den `sss` neu. Prüfen Sie anschließend im Log, ob es beim Starten des `sss` zu Fehlern gekommen ist. Wenn Sie die Fehlermeldungen aus Listing 17.33 im Log finden, weist das darauf hin, dass der `sss` über AppArmor abgesichert ist und dass Sie die Konfiguration von AppArmor noch anpassen müssen:



Oct 30 17:00:56 provider-dyn systemd[1]: Started System Security Services Daemon.

```
Oct 30 17:00:56 provider-dyn kernel: [11982.924658] audit: \
type=1400 audit(1604073656.245:24): apparmor="ALLOWED" \
operation="file_lock" profile="/usr/sbin/sss" \
name="/var/lib/sss/mc/passwd" pid=2216 comm="sss_nss" \
requested_mask="k" denied_mask="k" fsuid=0 ouid=0
```

```
Oct 30 17:00:56 provider-dyn kernel: [11982.925667] audit: \
type=1400 audit(1604073656.245:25): apparmor="ALLOWED" \
operation="file_lock" profile="/usr/sbin/sss" \
name="/var/lib/sss/mc/passwd" pid=2216 comm="sss_nss" \
requested_mask="k" denied_mask="k" fsuid=0 ouid=0
```

```
Oct 30 17:00:56 provider-dyn kernel: [11982.929154] audit: \
type=1400 audit(1604073656.245:26): apparmor="ALLOWED" \
operation="file_lock" profile="/usr/sbin/sss" \
name="/var/lib/sss/mc/group" pid=2216 comm="sss_nss" \
requested_mask="k" denied_mask="k" fsuid=0 ouid=0
```

```
Oct 30 17:00:56 provider-dyn kernel: [11982.929638] audit: \
type=1400 audit(1604073656.245:27): apparmor="ALLOWED" \
operation="file_lock" profile="/usr/sbin/sss" \
name="/var/lib/sss/mc/group" pid=2216 comm="sss_nss" \
requested_mask="k" denied_mask="k" fsuid=0 ouid=0
```

```
Oct 30 17:00:56 provider-dyn kernel: [11982.932265] audit: \
type=1400 audit(1604073656.245:28): apparmor="ALLOWED" \
operation="file_lock" profile="/usr/sbin/sss" \
name="/var/lib/sss/mc/initgroups" pid=2216 comm="sss_nss" \
requested_mask="k" denied_mask="k" fsuid=0 ouid=0
```

```
Oct 30 17:00:56 provider-dyn kernel: [11982.932370] audit: \
type=1400 audit(1604073656.245:29): apparmor="ALLOWED" \
operation="file_lock" profile="/usr/sbin/sss" \
name="/var/lib/sss/mc/initgroups" pid=2216 comm="sss_nss" \
requested_mask="k" denied_mask="k" fsuid=0 ouid=0
```

Listing 17.33 Fehler beim Start des »sss«

Passen Sie die Datei */etc/apparmor.d/usr/sbin.sssd* wie in Listing 17.34 an:

```

/etc/krb5.keytab k,
/etc/ldap/ldap.conf r,
/etc/libnl-3/classid r,
/etc/localtime r,
/etc/shells r,
/etc/sss/sss.conf r,
/etc/sss/conf.d/ r,
/usr/share/sss/cfg_rules.ini r,
/var/lib/sss/mc/passwd k,
/var/lib/sss/mc/group k,
/var/lib/sss/mc/initgroups k,
/etc/ssl/zertifikate/demoCA/cacert.pem r,
/etc/gss/mech.d/ r,

```

Listing 17.34 Anpassung von AppArmor für den »sss«

Anschließend starten Sie den AppArmor-Dienst und den sss neu. Prüfen Sie erneut, ob es zu Fehlern gekommen ist, und ergänzen Sie die Konfiguration entsprechend. Erst wenn beim Start des sss keine Fehler mehr auftreten, dürfen Sie mit den nächsten Schritten fortfahren.

17.5.1 Anlegen eines Testbenutzers

Um den sss auch testen zu können, benötigen Sie mindestens eine Gruppe und einen Benutzer im LDAP, die dann auch vom sss gefunden werden können.

Gruppen und Benutzer können Sie jetzt mit einer LDIF-Datei in die Objektdatenbank einspielen. In Listing 17.35 sehen Sie eine LDIF-Datei, mit der eine Gruppe und ein Benutzer erzeugt werden:

```

dn: cn=benutzer,ou=groups,dc=example,dc=net
objectClass: posixGroup
cn: benutzer
gidNumber: 10000

```

```

dn: uid=skania,ou=users,dc=example,dc=net
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
homeDirectory: /home/skania
loginShell: /bin/bash
uid: skania
cn: Stefan Kania
userPassword: ARGON2....

```

```
uidNumber: 10000
gidNumber: 10000
sn: Kania
givenName: Stefan
```

Listing 17.35 Anlegen von Testdatensätzen

Die beiden Datensätze spielen Sie jetzt wie gewohnt in den LDAP ein.

Mit dem Kommando `getent passwd <uid>` können Sie sich den gerade angelegten Benutzer anzeigen lassen. Äquivalent dazu können Sie sich mit dem Kommando `getent group <gid>` die gerade angelegte Gruppe anzeigen lassen.



Alle Benutzer und Gruppen anzeigen lassen

Wenn Sie `getent passwd` oder `getent group` ohne einen Benutzer- oder Gruppennamen ausführen, bekommen Sie keinen LDAP-Benutzer und keine LDAP-Gruppe angezeigt, denn das Enumerieren aller Benutzer und Gruppen ist in der Grundeinstellung des `sssd` deaktiviert. Der Grund ist der, dass die Auflistung aller Benutzer und Gruppen in einer großen Umgebung mit mehreren Hunderttausend Benutzern den Server zu stark belasten würde und das Ergebnis in den seltensten Fällen sinnvoll ist. Auch könnte ein Angreifer so sehr einfach die Namen aller Benutzerkonten erhalten.

Wollen Sie trotzdem alle Benutzer und Gruppen mit dem Kommando `getent` angezeigt bekommen, so ergänzen Sie die Konfiguration Ihres `sssd` um die folgende Zeile:

```
enumerate = true
```

Starten Sie anschließend den `sssd` neu, und führen Sie das Kommando `getent` erneut aus. Jetzt werden alle Benutzer und Gruppen immer angezeigt.

Damit ist die Grundkonfiguration des LDAP-Clients abgeschlossen. Wenn Sie jetzt alle Benutzer und Gruppen angelegt haben, können sich die Benutzer anmelden, und Sie können Rechte im Dateisystem an die LDAP-Benutzer und LDAP-Gruppen vergeben.

Auf allen Maschinen, die sich am LDAP authentifizieren sollen, können Sie den `sssd` identisch einrichten, das erleichtert dann auch die automatische Verteilung der Konfiguration.

17.6 Grafische Werkzeuge für die LDAP-Verwaltung

Natürlich ist es auf die Dauer nicht der beste Weg, nur mit den Kommandozeilentools zu arbeiten, um sich einen Überblick über die Struktur des LDAP-Baums zu verschaffen. Auch das Anlegen einzelner Objekte mithilfe von LDIF-Dateien ist auf die Dauer recht mühsam und mit Fehlern behaftet. Aber es gibt zum Glück verschiedene Werkzeuge, mit denen Sie

den DIT auch grafisch verwalten können. So können Sie bequem einzelne Objekte anlegen oder kleine Änderungen komfortabler vornehmen.

Größere Änderungen lassen sich zusätzlich immer noch über Skripte realisieren. Hier folgt eine kleine Auswahl an grafischen Werkzeugen:

► **LDAP Account Manager**

Der *LDAP Account Manager* (LAM) ist ein webbasiertes Werkzeug. Er wird auf einem Webserver installiert und kann verschiedene LDAP-Umgebungen verwalten. Das Werkzeug selbst müssen Sie nicht auf dem LDAP-Server installieren. Weitere Informationen und den Download finden Sie unter <http://lam.sourceforge.net>. Ein gute Anleitung zur Einrichtung des LAM finden Sie unter:

www.ldap-account-manager.org/static/doc/manual.pdf

Vom LAM gibt es zwei verschiedene Versionen: zum einen die freie Version, mit der Sie den LDAP-Server und auch einen Samba-4-Server verwalten können, zum anderen die kostenpflichtige Version, die weitere Möglichkeiten bereitstellt, wie beispielsweise die vollständige Verwaltung einer Kombination aus OpenLDAP und Kerberos. In Kapitel 14, »Kerberos«, gehen wir auf die Pro-Version und die Verbindung von OpenLDAP und Kerberos genauer ein. Hier in diesem Kapitel wird aber die freie Version zum Einsatz kommen. Ab der Version 7.9 können Sie jetzt auch die dynamische Konfiguration verwalten.

► **Apache Directory Studio**

Ein ebenfalls für die Installation auf dem lokalen Rechner geeignetes Verwaltungswerkzeug ist das *Apache Directory Studio*. Mit diesem Werkzeug können Sie nicht nur die Benutzerdatenbank, sondern auch die dynamische Konfiguration verwalten. Sie finden das Apache Directory Studio unter directory.apache.org/studio/. Apache Directory Studio basiert auf der Entwicklungsumgebung *Eclipse*, die Sie ebenfalls für die Administration Ihres LDAP-Servers nutzen können. Das Apache Directory Studio wurde aber um einige Module für die Verwaltung von Verzeichnisdiensten erweitert.

Natürlich kann die Liste der Werkzeuge hier nur einen ersten Überblick geben. Neben den hier beschriebenen Werkzeugen gibt es noch weitere, sowohl freie als auch kommerzielle.

17.7 Änderungen mit »ldapmodify«

Das Ändern von Objekteigenschaften über ein grafisches Frontend ist so lange gut und ausreichend, wie Sie nur einzelne Änderungen vornehmen wollen. Wenn Sie aber zum Beispiel die Telefonnummer für alle Objekte ändern wollen, ist diese Aufgabe mit einem grafischen Werkzeug doch sehr mühsam.

Deshalb gibt es das Kommando `ldapmodify`, mit dem Sie über LDIF-Dateien komplexe Änderungen an Objekten vornehmen können. Sie können die Änderungen sowohl interaktiv als auch skriptgesteuert durchführen.

17.7.1 Interaktive Änderung mit »ldapmodify«

Das Kommando `ldapmodify` können Sie auch direkt auf der Kommandozeile interaktiv verwenden, ohne vorher eine LDIF-Datei zu erzeugen. Dazu rufen Sie als `root` das Kommando `ldapmodify` wie in Listing 17.36 auf der Kommandozeile auf:

```
root@ldap01:~# ldapmodify -x -D "cn=admin,dc=example,dc=net" -W
Enter LDAP Password:
dn: uid=skania,ou=users,dc=example,dc=net
changetype: modify
add: description
description: Mein erster Benutzer
modifying entry "uid=skania,ou=users,dc=example,dc=net"
```

Listing 17.36 Interaktive Nutzung von »ldapmodify«

Nachdem Sie das Passwort eingegeben haben, geben Sie jede Zeile einzeln ein und bestätigen sie mit `↵`. Nach der letzten Zeile beenden Sie `ldapmodify` mit `Strg+D`. Daraufhin wird eine Erfolgsmeldung angezeigt. Das ist natürlich nur eine Lösung für einzelne Änderungen.

17.7.2 Änderungen über eine LDIF-Datei mit »ldapmodify«

Wollen Sie viele Änderungen durchführen, ist es einfacher, wenn Sie eine LDIF-Datei erzeugen und die Änderungen anschließend mit `ldapmodify` einlesen. Das hat auch den Vorteil, dass Sie bei einem Syntaxfehler einfach die Datei ändern können und nicht alles wieder von Hand neu eingeben müssen. In diesem Schritt soll für die Container `ou=users` und `ou=groups` eine Beschreibung als Attribut hinzugefügt werden. Dazu erstellen Sie eine LDIF-Datei mit dem Inhalt aus Listing 17.37:

```
dn: ou=users,dc=example,dc=net
changetype: modify
replace: description
description: Container fuer alle Benutzer

dn: ou=groups,dc=example,dc=net
changetype: modify
add: description
description: Container fuer alle Gruppen
```

Listing 17.37 LDIF-Datei für Modifikationen an mehreren Objekten

Die Leerzeile zwischen den beiden Änderungen ist wichtig, denn dadurch kann `ldapmodify` erkennen, dass es sich um Änderungen für mehrere Objekte handelt. Jedes weitere Objekt wird immer durch eine Leerzeile getrennt in die Datei geschrieben. Nachdem Sie die Datei erstellt haben, können Sie die Änderungen mit `ldapmodify` so wie in Listing 17.38 einspielen:

```
ldapmodify -x -D "cn=admin,dc=example,dc=net" -W -f modify.ldif
Enter LDAP Password:
modifying entry "ou=users,dc=example,dc=net"
```

```
modifying entry "ou=groups,dc=example,dc=net"
```

Listing 17.38 Einspielen der Änderungen

Nach der Änderung können Sie die Objekte sowohl über die Kommandozeile mit dem Kommando `ldapsearch` als auch mit dem LAM überprüfen.

Wollen Sie an ein und demselben Objekt mehrere Änderungen gleichzeitig durchführen, können Sie das auch über eine *LDIF*-Datei realisieren. Listing 17.39 zeigt ein Beispiel für eine *LDIF*-Datei, mit der bei einem Objekt mehrere Attribute geändert werden:

```
dn: uid=skania,ou=users,dc=example,dc=net
changetype: modify
replace: description
description: Eine neue Beschreibung
-
add: mobile
mobile: 177 12345676
```

Listing 17.39 LDIF-Datei für mehrere Änderungen an einem Objekt

Wichtig sind dabei die Trennlinien mit dem Minuszeichen: Daran erkennt `ldapmodify`, dass die nachfolgenden Änderungen an ein und demselben Objekt durchgeführt werden sollen. Die Änderungen werden genau wie im Beispiel vorher mit `ldapmodify` eingespielt.

Jetzt können Sie natürlich auch hingehen und den Objektnamen, der geändert werden soll, sowie die Werte der Attribute über ein Skript aus Dateien auslesen und damit viele Benutzer gleichzeitig ändern.

17.8 Absichern des LDAP-Baums mit ACLs

Inzwischen werden in unserem Beispiel alle Zugriffe auf den Server verschlüsselt durchgeführt. Auch haben wir durch die Grundkonfiguration sämtliche anonymen Zugriffe mittels `ldapsearch -x` unterbunden. Jetzt ist es aber notwendig, dass Sie die Zugriffe Schritt für Schritt wieder erlauben. Dabei sollten Sie immer vorher überlegen, welcher Benutzer welche Zugriffsrechte benötigt. Für die Steuerung der Zugriffsrechte werden ACLs verwendet. Mithilfe der ACLs können Sie Zugriffe auf ganze Zweige des DIT, auf einzelne Objekte oder einzelne Attribute steuern. Nur eine gut geplante und eingerichtete Struktur des DIT macht die Verwaltung der ACLs einfach und übersichtlich. Die Möglichkeiten, um Zugriffe mit ACLs zu regeln, sind sehr vielfältig. Auf alle Möglichkeiten kann hier im Buch nicht eingegan-

gen werden, doch soll an einigen Beispielen die Arbeitsweise der ACLs beschrieben werden. Gerade dadurch, dass Sie auch reguläre Ausdrücke in ACLs verwenden können, haben Sie weitreichende Möglichkeiten, die ACLs sehr gut an Ihre eigenen Belange anzupassen.



Dabei sollten Sie darauf achten, dass Sie nicht zu viele ACLs einsetzen, denn ACLs können sich negativ auf die Zugriffszeiten auf den DIT auswirken – vor allem dann, wenn Sie sehr viele reguläre Ausdrücke verwenden.

Wie und wo Sie die ACLs verwalten, hängt davon ab, ob Sie die dynamische oder die statische Konfiguration für Ihren LDAP-Server verwenden. Wir werden hier nur noch auf die dynamische Einrichtung der ACLs eingehen.

Eine sehr umfangreiche und ausführliche Hilfe zum Thema ACLs finden Sie auch in der Manpage *slapd.access*. Grundsätzlich haben alle Regeln den Aufbau wie in Listing 17.40:

```
access to <what>
  by <who> <access> [<control>]
```

Listing 17.40 Grundstruktur der ACLs

Eine ACL ist für den LDAP-Server immer ein Einzeiler. Da Sie aber Rechte über ACLs an verschiedene Objekte vergeben können, würden die Einträge sehr unübersichtlich, wenn Sie alles in eine Zeile schreiben würden. Aus diesem Grund können Sie die *by*-Zeilen eingerückt unter die Zeile schreiben die mit *access* beginnt schreiben. Da die *access*-Zeile immer linksbündig stehen muss, erkennt der *slapd* dann, welche Zeilen zu einer ACL gehören. Schauen wir uns die einzelnen Felder einmal genauer an:

Das »what«-Feld

Hier legen Sie fest, an welchem Objekt Sie Rechte vergeben wollen. In das Feld können Sie ein einzelnes Objekt, einen ganzen Teilbaum oder nur ein einzelnes Attribut eintragen. Auch können Sie mehrere Objekte an unterschiedlichen Stellen im DIT durch reguläre Ausdrücke abbilden. Objekte werden hier mit Ihrem DN angegeben. Neben dem DN können Sie hier auch noch einen *dnstyle* angeben, der die Suchtiefe festlegt. Den *dnstyle* können Sie später auch im *who*-Feld nutzen; die Bedeutung der einzelnen Styles ist dort identisch. Folgende Styles stehen Ihnen zur Verfügung:

► exact, base

Diese beiden Styles beschreiben genau ein Objekt. Wenn Sie also genau für ein Objekt Rechte vergeben wollen, dann nutzen Sie diesen Style. Wenn Sie das Objekt mit *dn.exact=ou=users,dc=example,dc=net* angeben, werden sich die Berechtigungen der *who*-Zeilen nur genau auf das Objekt beziehen – in diesem Fall nur auf die OU, in der die Benutzer abgelegt sind, nicht aber auf die Benutzerobjekte in der OU.



Auch DN ohne Style nutzbar

Wenn Sie keinen Style angeben wie in `dn=ou=users,dc=example,dc=net`, dann betrifft diese ACL auch nur dieses eine Objekt und keine untergeordneten Objekte.

- ▶ `one`
Mit `dn.one=ou=users,dc=example,dc=net` vergeben Sie Sie Berechtigungen auf das hier angegebene Objekt sowie auf alle Objekte, die sich direkt in der OU befinden. Sollte es in der OU weitere untergeordnete OUs geben, wären die Rechte in den OUs nicht mehr relevant.
- ▶ `sub`
Wollen Sie Rechte an einem ganzen Teilbaum und an allen aktuellen und zukünftigen Objekten vergeben, egal in welcher OU-Tiefe sich die Objekte befinden, dann verwenden Sie den Style `dn.sub=ou=users,dc=example,dc=net`.
- ▶ `children`
Im Style `children` vergeben Sie die Rechte an allen untergeordneten Objekten, nicht aber an dem Objekt das Sie im `what`-Feld beschrieben haben. Ein `dn.children=ou=users,dc=example,dc=net` würde zwar allen Objekten in den `who`-Zeilen Rechte an den untergeordneten Objekten von `ou=users` geben, nicht aber an der OU selbst.

Das »who«-Feld

Über das `who`-Feld legen Sie fest, wer Rechte an dem im `what`-Feld beschriebenen Objekt erhält. Neben den Objekten und den vorher beschriebenen Styles gibt es noch weitere spezielle Einträge, die Sie hier verwenden können:

- ▶ `anonymous`
Mithilfe dieses Eintrags können Sie anonyme Anfragen an OpenLDAP verbieten oder nur bestimmte Rechte gewähren. Oft werden Sie `anonymous` dazu nutzen, um Rechte während der Anmeldung zu gewähren. Einen Eintrag in den ACLs werden Sie jetzt schon in Ihrer Konfiguration finden, und zwar den, der den Zugriff auf das Passwort für anonyme Benutzer auf die Authentifizierung einschränkt.
- ▶ `users`
Der Eintrag `users` ist das genaue Gegenteil von `anonymous`. Hier gewähren Sie denjenigen Benutzern Rechte, die sich zuvor gegen den LDAP authentifiziert haben.
- ▶ `self`
Verwenden Sie `self`, um einem Benutzer Rechte am eigenen Objekt zu geben.
- ▶ `group`
Sie können auch Rechte an Mitglieder einer Gruppe vergeben, nur muss es sich bei der Gruppe um eine Gruppe handeln, in der die Mitglieder mit ihrem DN eingetragen

sind. Bei den Gruppen handelt es sich um die Objekte `groupOfURLs`, `groupOfNames` und `groupOfUniqueNames`. Die Verwendung von POSIX-Gruppen ist hier nicht möglich.

▶ *

Den Stern können Sie als Wildcard einsetzen. Der Stern steht für alles.

Die »access«-Einträge

Jetzt fehlen noch die Rechte, die Sie an die Objekte im `who`-Feld vergeben können. In der folgenden Auflistung sehen Sie alle Rechte und ihre Beschreibung. Ein Recht aus der Aufzählung schließt immer die Rechte des vorherigen Punktes ein.

▶ none

Wenn Sie sicher sein wollen, dass ein Objekt aus einem `who`-Feld keine Rechte besitzen soll, dann verwenden Sie `none`, zum Beispiel `by anonymous none`. Wenn ein Benutzer auf die Berechtigung `none` trifft und er dadurch keinen Zugriff erhält, bekommt er auch keinen Hinweis auf die fehlenden Berechtigungen.

▶ disclose

Das Recht `disclose` sorgt genau wie `none` dafür, dass für den Zugriff auf das Objekt im `what`-Feld keine Rechte vergeben werden. Der Unterschied zu `none` ist der, dass bei einem Zugriff hier eine Meldung über die fehlenden Rechte ausgegeben wird.

▶ auth

Alle Attribute oder Objekte, an denen das Recht `auth` vergeben wird, können nur zur Authentifizierung verwendet werden. Eine Anzeige, eine Suche oder ein Vergleich der Attribute ist nicht möglich.

▶ compare

Vergeben Sie an einem Attribut das Recht `compare`, kann der Inhalt des Attributs mit einem Wert verglichen werden. Als Ergebnis gibt es entweder ein `True` oder ein `False`. Der Inhalt des Attributs wird nicht angezeigt.

▶ search

Wenn Sie zum Beispiel in einem Teilbaum nach allen Objekten suchen, die bei einem bestimmten Attribut einen bestimmten Inhalt besitzen, dann benötigen Sie das Recht `search`. Der Inhalt des Attributs wird aber auch nicht angezeigt.

▶ read

Mit dem Recht `read` kann der Inhalt eines Attributs angezeigt werden.

▶ write,add,delete

Vergeben Sie das Recht `write`, können Attribute oder Objekte verändert, hinzugefügt und gelöscht werden. Das Recht können Sie noch einschränken, indem Sie das Recht `add` oder `delete` vergeben. Wenn Sie zum Beispiel einem Benutzer an einer OU das Recht `add` geben, kann er neue Objekte in der OU anlegen, diese ändern, aber nicht löschen. Vergeben Sie

an einer OU das Recht `delete`, kann der Benutzer in der OU bestehende Objekte löschen, aber nicht ändern oder neue Objekte anlegen.

- ▶ `manage`
Über das Recht `manage` geben Sie einem Benutzer administrative Rechte. Das Recht sollte sehr sparsam eingesetzt werden. In den meisten Fällen reicht das Recht `write`, um Objekte im DIT zu verwalten.

Die »control«-Einträge

Eine nicht ganz unwichtige Möglichkeit, die Abarbeitung der ACLs zu steuern, ist über das `control`-Feld gegeben. Über dieses Feld können Sie die Abarbeitung weiterer Regeln erzwingen. Die folgenden drei `control`-Einträge stehen Ihnen zur Verfügung:

- ▶ `stop`
Der Control-Parameter `stop` ist grundsätzlich der Standardwert am Ende der letzten `by`-Zeile und sorgt dafür, dass bei einem *Match* der Regel die Liste sofort verlassen wird.
- ▶ `continue`
Verwenden Sie `continue` am Ende einer `by`-Zeile, werden eventuell vorhandene weitere Zeilen innerhalb dieser ACL noch geprüft. So kann ein Benutzer eventuell mehr Rechte erhalten, wenn er in zwei Gruppen Mitglied ist, beide Gruppen in die ACL eingetragen wurden und die zweite Gruppe in der Liste mehr Rechte hat.
- ▶ `break`
Mit dem Eintrag `break` am Ende der letzten `by`-Zeile sorgen Sie dafür, dass alle folgenden Regeln auch noch geprüft werden. Diese Option benötigen Sie später noch, um einem Replikationsbenutzer Rechte an allen Objekten zu geben.

Über die `access`-Einträge können Sie die einzelnen Rechte noch feiner vergeben oder bestimmte Berechtigungen wieder entfernen. Wir werden hier im Buch nicht weiter auf die Möglichkeiten mit den `access`-Einträgen eingehen, wenn Sie hierzu weitere Informationen benötigen, schauen Sie in die Manpage zu `slapd.access`.

17.9 Grundlegende ACLs

Bei der dynamischen Konfiguration werden die ACLs, wie schon zuvor alle anderen Änderungen an der Konfiguration, mittels LDIF-Dateien eingespielt. Jede erfolgreiche Änderung der Konfiguration ist sofort wirksam. Ein Neustart des LDAP-Dienstes ist nicht notwendig um die ACL wirksam werden zu lassen.

Während der Grundkonfiguration beim Installieren des OpenLDAP werden bereits ACLs erzeugt. Diese können Sie sich mit einem Blick in die Konfiguration ansehen. In Listing 17.41 sehen Sie das Kommando zur Anzeige der Konfiguration und den entsprechenden Ausschnitt:

```
root@ldap01:~# ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config -LLL | less
[...]
dn: olcDatabase={-1}frontend,cn=config
olcAccess: {0}to *
  by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
  by * break
olcAccess: {1}to dn="" by * read
olcAccess: {2}to dn.base="cn=subschema" by * read

dn: olcDatabase={0}config,cn=config
[...]
dn: olcDatabase={0}config,cn=config
objectClass: olcDatabaseConfig
olcDatabase: {0}config
olcAccess: {0}to *
  by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
  by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net write
  by dn.exact=uid=repl-user,ou=users,dc=example,dc=net read
[...]

dn: olcDatabase={1}monitor,cn=config
[...]
olcAccess: {0}to dn.subtree="cn=monitor"
  by dn.exact=cn=admin,cn=config read
  by dn.exact=cn=admin,dc=example,dc=net read
  by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net read
[...]

dn: olcDatabase={2}mdb,cn=config
[...]
olcAccess: {0} to *
  by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
  by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net write
  by dn.exact=uid=sssd-user,ou=users,dc=example,dc=net read
  by dn.exact=uid=repl-user,ou=users,dc=example,dc=net read
  by * break
olcAccess: {1} to dn.exact="dc=example,dc=net" by users read
olcAccess: {2} to attrs=userPassword
  by anonymous auth
```



```

by self write
by * none
[...]
```

Listing 17.41 ACLs nach der Installation

Da die dynamische Konfiguration in mehrere Datenbanken oder Datenbankbereiche unterteilt ist, finden Sie ACLs am Anfang an vier verschiedenen Stellen in der Konfiguration:

- ▶ dn: olcDatabase={-1}frontend,cn=config
Hier finden Sie die ACLs, die global, also für alle Datenbanken des LDAP-Servers, Gültigkeit haben.
- ▶ dn: olcDatabase={0}config,cn=config
Da sich die dynamische Konfiguration in einer eigenen Datenbank befindet, können Sie auch hierfür Berechtigungen vergeben.
- ▶ dn: olcDatabase={1}monitor,cn=config
Das ist der Bereich für die Informationen des Monitorings. Vergeben Sie auf jeden Fall immer auch eine ACL für die Monitoring-Datenbank, da sonst jeder anonym auf die Informationen der Monitoring-Datenbank zugreifen kann.
- ▶ dn: olcDatabase={2}mdb,cn=config
Das ist der Bereich, in dem Sie wahrscheinlich die meisten Änderungen durchführen werden, denn hier vergeben Sie die Berechtigungen für den Zugriff auf Ihre Objektdatenbank.

Wenn Sie unsere Grundkonfiguration nutzen, haben wir schon ein paar ACLs eingepflegt:

- ▶ dn: olcDatabase={-1}frontend,cn=config
Hier sorgen wir dafür, dass auch anonym die *root* des DIT dn= von jedem gelesen werden kann. Wenn Sie diese ACL nicht setzen, ist das wie eine verschlossene Haustür. Keiner kommt rein, auch wenn Rechte auf einer tieferliegenden Ebene vorhanden wären. Auch das Schema muss von jedem, der auf dem LDAP zugreift, gelesen werden können. Da diese Berechtigungen für alle auf dem LDAP-Server verwalteten Datenbanken benötigt werden, wurden die ACLs in den globalen Bereich eingetragen.
- ▶ dn: olcDatabase={1}monitor,cn=config
Auch für das Monitoring gibt es eine eigene Datenbank, für die Sie die Zugriffsrechte verwalten müssen. Eventuell wollen Sie später einen Benutzer anlegen, der nur hier Zugriff hat und die Daten dann in Ihre Monitoringsoftware übernehmen kann. Hier wird auf jeden Fall nur das Leserecht benötigt.
- ▶ dn: olcDatabase={2}mdb,cn=config
Wie Sie sehen, haben wir hier schon ein paar Einträge mehr. In der ersten ACL mit der Nummer 0 steuern wir den Zugriff auf alle Objekte und alle Attribute. Diese Regel wird immer als erste abgearbeitet, sodass es für diese Benutzer beim Zugriff auf gar keinen Fall zu ungewollten Einschränkungen kommt. Dort haben wir auch schon den Benutzer

uid=ldap-admin,ou=users,dc=example,dc=net mit der Berechtigung `write` und den Benutzer uid=repl-user,ou=users,dc=example,dc=net mit der Berechtigung `read` angelegt. Beide Benutzer sind noch nicht in der Objektdatenbank angelegt, trotzdem können Sie die ACL schon erzeugen. Es wird bei ACLs nie geprüft, ob ein entsprechendes Objekt für den Eintrag in der ACL vorhanden ist. Ganz wichtig an dieser Stelle ist die Zeile `by * break`: Wenn ein anderer als die hier aufgelisteten Benutzer zugreift, wird über die folgenden ACLs geprüft, ob der Benutzer doch an bestimmten Objekten Rechte besitzt. Wenn Sie diese Zeile vergessen, wird dort der Standardwert `by * none` gesetzt und niemand könnte auf die untergeordneten Objekte zugreifen, auch wenn Sie im Verlauf der Liste aller ACLs noch Rechte vergeben haben. Denn hier gilt: Sobald eine ACL auf einen Zugriff passt, wird die Prüfung der weiteren ACLs abgebrochen.

- ▶ Mit der folgenden ACL wird für alle authentifizierten Benutzer das Leserecht an der obersten Ebene unseres LDAP-Baums vergeben.
- ▶ Zum Abschluss folgt eine ACL, die den Zugriff auf die Passwörter der Benutzer steuert. Hier sehen Sie eine ACL, die sich nicht auf ein ganzes Objekte sondern nur auf ein einzelnes Attribut bezieht. Diese Regel setzt die Rechte nur für anonyme Zugriffe und für den Zugriff von Benutzern auf ihr eigenes Passwort. Unser *ldap-admin* kann die Passwörter der Benutzer ändern, denn er erhält den Zugriff über die erste ACL `olcAccess: {0} to *`.



Achten Sie genau auf die Reihenfolge

Immer wenn Sie ACLs ergänzen, ändern oder in der Reihenfolge verändern, müssen Sie darauf achten, dass die Nummerierung passend ist. Jede Änderung, die Sie einspielen, ist sofort ohne Neustart des *slapd* wirksam. Da Sie nicht wie bei der statischen Konfiguration eine ACL auskommentieren können, wird beim Ändern auch immer die bestehende ACL überschrieben oder gelöscht. Auch die Reihenfolge ist nicht ohne Weiteres wiederherstellbar. Besser ist es, wenn Sie hier zweimal prüfen. Bei größeren Änderungen können Sie die Anzeige der Konfiguration auch in eine Datei umleiten. So sind Sie wenigstens in der Lage, den alten Stand wiederherstellen zu können.

17.10 Der neue LDAP-Admin

Bis zu diesem Zeitpunkt haben Sie alle Änderungen an der Objektdatenbank mit dem *rootdn* oder mit dem SASL-Mechanismus `EXTERNAL` durchgeführt. Das soll sich jetzt ändern. Bei dem *rootdn* handelt es sich um den wichtigsten Benutzer, der immer alle Änderungen an den Objekten vornehmen kann. Er kann in seinen Rechten nicht durch ACLs oder andere Mechanismen beschnitten werden. Aus diesem Grund ist es keine gute Idee, die Administration mit dem *rootdn* durchzuführen. Der SASL-Mechanismus kann nur lokal auf dem LDAP-Ser-

ver genutzt werden, nicht aber über das Netzwerk. Besser, Sie legen sich einen Benutzer als `simpleSecurityObject` an, mit dem Sie alle Änderungen durchführen. Für den Fall, dass dieses Objekt kompromittiert würde und das Passwort bekannt würde, könnten Sie dieses Objekt einfach löschen und ein neues anlegen. Auch hätten Sie so die Möglichkeit, das Objekt in seinen Rechten zu beschränken und die Administration der Objekte auf mehrere Personen mit unterschiedlichen Zugriffsobjekten zu verteilen.

Bei der Verwaltung der dynamischen Konfiguration kommt ein ganz anderes Problem zum Tragen. Zum einen lässt sich der SASL-Mechanismus nur lokal auf dem LDAP-Server verwenden und zum anderen kann der Mechanismus nur vom Benutzer `root` genutzt werden. Warum verwenden wir ein `simpleSecurityObject` und kein POSIX-Account? Der Grund ist der, dass sich niemand mit einem `simpleSecurityObject` lokal an einem LDAP-Client anmelden kann. Dieses Objekt kann nur dazu genutzt werden, im LDAP zu suchen. Ein Objekt dieser Objektklasse benötigt lediglich einen DN und ein Passwort.

17.10.1 Anlegen der Objekte

Das Objekt wird in der Objektdatenbank über eine LDIF-Datei angelegt (siehe Listing 17.42):

```
dn: uid=ldap-admin,ou=users,dc=example,dc=net
objectClass: account
objectClass: simpleSecurityObject
objectClass: top
uid: ldap-admin
userPassword: ARGON2....
```

Listing 17.42 LDIF-Datei für die Admin-Benutzer

Da die Berechtigungen für den Benutzer bereits in der Grundkonfiguration eingetragen wurden, brauchen Sie an den ACLs keine Änderungen mehr vorzunehmen. Testen Sie jetzt den Zugriff mit dem Kommando `ldapsearch -x -D uid=ldap-admin,ou=users,dc=example,dc=net -w -LLL`: Sie sehen alle Objekte und bei den Benutzerobjekten auch die Passwörter.

Nachdem Sie das Objekt angelegt haben, können Sie eine LDIF-Datei anlegen um einen neuen Benutzer oder eine neue Gruppe im LDAP zu erzeugen. Da für den `ldap-admin` auch eine ACL in der Konfigurationsdatenbank vorhanden ist, können Sie mit dem Benutzer auch die Konfiguration ändern.

Entfernen der »rootdn«-Passwörter

Wenn Sie verhindern wollen, dass weiterhin Änderungen mit dem `rootdn` durchgeführt werden, können Sie jetzt das Passwort für den `rootdn` aus beiden Datenbanken entfernen.



17.11 Absichern der Passwörter

Eine der wichtigsten Regeln ist die, mit der Sie die Passwörter der Benutzer vor unerlaubtem Zugriff geschützt werden. Nur der *ldap-admin*, eine bestimmte Gruppe von Benutzern und der Benutzer selbst sollen das Recht haben, Passwörter ändern zu können.

Um einer Gruppe Rechte geben zu können, benötigen Sie eine Gruppe, in der der vollständige DN des Benutzers als Mitglied eingetragen ist, denn sonst lässt sich der Benutzer nicht eindeutig ermitteln und es können keine Rechte vergeben werden. Damit fällt die »normale« POSIX-Gruppe als Möglichkeit aus. Denn dort wird lediglich der Name als Mitglied eingetragen. Sie benötigen ein anderes Objekt. Zur Auswahl stehen Ihnen die folgenden drei Objekte:

- ▶ `groupOfNames`
Bei der `groupOfNames` werden alle Mitglieder der Gruppe mit ihrem vollständigen DN in dem Attribut `members` abgelegt. Die Gruppe muss immer mindestens ein Mitglied haben.
- ▶ `groupOfUniqueNames`
Bei der `groupOfUniqueNames` werden alle Mitglieder mit ihrem vollständigen DN im Attribut `uniqueMember` abgelegt. Die Gruppe muss mindestens ein Mitglied haben. Im Unterschied zur Gruppe `groupOfNames` muss der Name hier eindeutig sein. Haben Sie eine Mitarbeiterin »Petra Meier« als `cn=pmeier,ou=users,dc=example,dc=net` angelegt sowie einer Gruppe hinzugefügt und löschen Sie später das Objekt, ohne es aus der Gruppe zu entfernen, bleibt der Eintrag erhalten – es sei denn, Sie haben das Overlay `refint` eingerichtet. Legen Sie jetzt einen neuen Benutzer mit dem Namen »Peter Meier« mit dem DN `cn=pmeier,ou=users,dc=example,dc=net` an und wollen Sie ihn zur Gruppe hinzufügen, könnten Sie nicht mehr sagen, welcher Eintrag zu welchem Objekt gehört. Bei der Verwendung von `groupOfUniqueNames` wäre das nicht möglich, da auf Eindeutigkeit des DNs geprüft wird.
- ▶ `groupOfURLs`
Auch in `groupOfURLs` können die Mitglieder mit ihrem vollständigen DN abgelegt werden. Diese Art der Gruppe wird aber hauptsächlich für die Verwaltung von dynamischen Gruppen verwendet.

Im Folgenden soll jetzt ein `groupOfNames`-Objekt genutzt werden, da die Eindeutigkeit hier keine Rolle spielt. Wie schon erwähnt, lässt sich die Gruppe nur anlegen, wenn bereits beim Anlegen ein `member` mit angegeben wird. Das Schöne bei der Verwendung von `groupOfNames` ist, dass Sie einfach einen DN angeben können. Es wird nicht geprüft, ob der Benutzer existiert. So können Sie einen *dummy-user* angeben, der nie entfernt wird. Auf diese Weise habe Sie immer einen entsprechenden Eintrag in der Gruppe, auch wenn Sie einmal aus Versehen alle echten DNs gelöscht haben. Es hat noch einen weiteren Vorteil: Wenn Sie später das Overlay `refint` einrichten, kann dadurch nicht automatisch die Liste der Mitglieder leer werden. In Listing 17.43 sehen Sie die LDIF-Datei, mit der Sie die Gruppe eintragen können:

```
dn: cn=pw-set,ou=groups,dc=example,dc=net
cn: pw-set
member: uid=dummy-user,ou=users,dc=example,dc=net
objectClass: groupOfNames
objectClass: top
```

Listing 17.43 Anlegen der Gruppe für Passwortänderungen

Jetzt fehlt uns nur noch die Anpassung der ACL für den Zugriff auf die Passwörter. In Listing 17.44 sehen Sie die LDIF-Datei für die Anpassung der ACL:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
delete: olcAccess
olcAccess: {2}
-
add: olcAccess
olcAccess: {2}to attrs=userPassword
    by self write
    by group/groupOfNames/member=cn=pw-set,ou=groups,dc=example,dc=net write
    by anonymous auth
    by * none
-
add: olcAccess
olcAccess: {3}to attrs=shadowLastChange
    by self write
    by group/groupOfNames/member=cn=pw-set,ou=groups,dc=example,dc=net write
    by * read
```

Listing 17.44 ACL für die Gruppenberechtigung

Die zusätzliche ACL auf das Attribut `shadowLastChange` ist notwendig, da bei einer Änderung des Passworts auch das Datum der letzten Änderung geschrieben werden muss.

17.12 ACLs mit regulären Ausdrücken

Um das Thema der ACLs zu vervollständigen, fehlt jetzt noch ein Blick auf ACLs mit regulären Ausdrücken. Denn oft ist es nicht möglich, jeden Bereich einzeln mit einer ACL abzudecken: Der Aufwand wäre einfach zu hoch. Stellen Sie sich vor, Sie wollen allen Mitarbeitern im Unternehmen die Möglichkeit geben, ein eigenes Adressbuch im LDAP anzulegen, und Sie haben mehrere Tausend Benutzer. Jetzt können Sie nicht individuelle ACLs für jeden Benutzer vergeben, sondern dieser Vorgang sollte möglichst mit einer einzigen ACL realisierbar sein. Für diese Fälle gibt es die Möglichkeit, reguläre Ausdrücke in den ACLs zu nutzen.

Um ein Beispiel für die ACLs zu bekommen, sehen Sie in Listing 17.45 einen Ausschnitt einer LDIF-Datei, mit der ein kleiner Baum mit verschiedenen OUs und Benutzern angelegt wird – die vollständige Datei finden Sie im Downloadmaterial. Jeder Benutzer bekommt eine eigene OU unterhalb seines Objekts, in dem er seine eigenen Einträge verwalten kann. Alle anderen Benutzer sollen keinen Zugriff erhalten.

[...]

```
dn: cn=u2 Prod,ou=users,ou=Produktion,dc=example,dc=net
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
loginShell: /bin/bash
homeDirectory: /home/u2-prod
uid: u2-prod
cn: u2 Prod
userPassword:: e1NTSEF9bUdNcytBMzlicExENFB0dUxOL1ozMnMxZ3cxeVVEbE4=
uidNumber: 10002
gidNumber: 10000
sn: Prod
givenName: u2
dn: cn=prod al,ou=users,ou=Produktion,dc=example,dc=net
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
loginShell: /bin/bash
homeDirectory: /home/prod-al
uid: prod-al
cn: prod al
userPassword:: e1NTSEF9cGRGY3ZGNHJpQnMrZ2w5bTkvbDVMM2tnbERodVNtZHg=
uidNumber: 10003
gidNumber: 10000
sn: al
givenName: prod
employeeType: Abteilungsleiter
```

[...]

```
dn: ou=Adressen,cn=u1 Prod,ou=users,ou=Produktion,dc=example,dc=net
ou: Adressen
objectClass: organizationalUnit
objectClass: top
```

```
dn: ou=Adressen,cn=u2 Prod,ou=users,ou=Produktion,dc=example,dc=net
ou: Adressen
objectClass: organizationalUnit
objectClass: top
```

Listing 17.45 Zusätzliche Objekte für einen Adressbuchtest

Bei den letzten sechs Objekten im Listing handelt es sich um die OUs, in denen die Benutzer später ihre eigenen Adressen verwalten können sollen. Nachdem Sie die Objekte angelegt haben, fehlt jetzt noch die entsprechende ACL. In Listing 17.46 sehen Sie die ACL:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
add: olcAccess
olcAccess: {4}to dn.regex=ou=adressen,cn=(.+),ou=users,ou=(.+),dc=example,dc=net$
    by dn.regex=^cn=$1,ou=users,ou=$2,dc=example,dc=net$ write
    by * none
```

Listing 17.46 ACL für Adressbücher (statisch)

Die ACL gewährt den Zugriff auf die `ou=adressen` für jeden beliebigen `cn=(.+)` in jeder `ou=users`, unterhalb einer beliebigen Abteilung `ou=(.+)` mit dem Suffix `dc=example,dc=net`.

Schreibenden Zugriff hat nur der `cn=$1`, der im ersten regulären Ausdruck ermittelt wurde, und zwar in jeder `ou=users`, die unterhalb der Abteilung `ou=$2` die im Suffix `dc=example,dc=net` liegt. Wichtig ist hier, dass Sie den Anfang und das Ende des regulären Ausdrucks begrenzen, da sonst auch untergeordnete Objekte dieses Recht hätten.

Wenn Sie jetzt eine Suche mit einem der vorher erstellten Benutzer durchführen, werden Sie das Ergebnis aus Listing 17.47 erhalten:

```
root@ldap01:~# ldapsearch -x -D "cn=u1 prod,ou=users,ou=produktion,\
dc=example,dc=net" -W -LLL
Enter LDAP Password:
dn: dc=example,dc=net
objectClass: domain
objectClass: dcObject
dc: example
```

Listing 17.47 Erster Zugriffsversuch

Sie sehen: Der Benutzer sieht nur die oberste Ebene. Sein Adressbuch und sein Objekt kann er nicht sehen, auch die Objekte seiner Abteilung werden nicht aufgelistet. Aber warum?

Der Benutzer könnte zwar auf sein Adressbuch zugreifen, aber er kommt nicht in die OU der Abteilung, denn dort hat er keine Rechte. Damit die Mitarbeiter aus den zwei Abteilungen

Zugriff auf ihre Objekte bekommen, fehlen noch weitere ACLs für jede Abteilung. Diese ACLs sehen Sie in Listing 17.48:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
add: olcAccess
olcAccess: {5}to dn.sub=ou=verwaltung,dc=example,dc=net
    by dn.children=ou=users,ou=verwaltung,dc=example,dc=net read
    by * none
-
add: olcAccess
olcAccess: {6}to dn.sub=ou=produktion,dc=example,dc=net
    by dn.children=ou=users,ou=produktion,dc=example,dc=net read
    by * none
```

Listing 17.48 Alle benötigten ACLs

Testen Sie den Zugriff erneut, und Sie werden sehen: Jetzt klappt es, und Sie sehen das erwartete Ergebnis. Achten Sie hier auf die Reihenfolge der ACLs. Die ACL für Objekte tiefer im DIT, also die Adressbücher, steht über der ACL für die Abteilung. Würden Sie die ACLs tauschen, würde beim Zugriff auf ein Adressbuch schon die Regel für die Abteilung greifen. Das bedeutet: Alle Mitarbeiter der Abteilung könnten alle Adressbücher aller Mitarbeiter der Abteilung lesen. Die beiden Regeln für die Abteilungen schreien quasi nach dem Einsatz eines regulären Ausdrucks, da sie sich nur durch den Namen der Abteilung unterscheiden und sie sich auch noch auf derselben Ebene befinden. Aber da die Abarbeitung von regulären Ausdrücken in ACLs sehr aufwendig ist, ist es oft sinnvoller, einzelne ACLs zu schreiben.

In unserem Beispiel wird wohl so schnell keine neue Abteilung hinzugefügt werden, die Ebene ist relativ statisch. Bei den Mitarbeitern in den Abteilungen sieht es schon anders aus, da kommen immer wieder neue hinzu oder werden versetzt oder verlassen das Unternehmen. In diesem Fall wäre es eine Sisyphusarbeit, für jeden Mitarbeiter eine eigene ACLs zu erstellen. Was Ihnen dieses Beispiel noch vermitteln will, ist, dass Sie immer prüfen müssen, ob es reicht, eine ACL auf das Objekt selbst anzulegen, oder ob Sie noch ACLs für übergeordnete Objekte berücksichtigen müssen.

Sollten die ACLs nicht funktionieren oder falls Sie alle gerade erstellten ACLs wieder entfernen wollen, um sie noch einmal zu ändern und sie erneut einzuspielen, achten Sie darauf, dass Sie die ACLs in umgekehrter Reihenfolge löschen. Wenn Sie hier die vierte ACL als Erstes löschen, führt das nämlich dazu, dass alle ACLs um einen Platz aufrücken. Das bedeutet, aus der 5 würde eine 4 und aus der 6 eine 5. Wenn Sie dann die fünfte ACL löschen, würde das dazu führen, dass Sie die ACL löschen die ursprünglich die Nummer 6 hatte. Am Ende würden entweder nicht alle ACLs gelöscht – oder viele schlimmer – Sie würden eventuell die falschen ACLs löschen. Um alle gerade erstellten ACLs fehlerfrei zu löschen, erstellen Sie sich

eine LDIF-Datei mit dem Inhalt aus Listing 17.49. Immer wenn Sie in Zukunft neue Objekte in Ihren DIT aufnehmen, achten Sie auf die ACLs!

```
dn: olcDatabase={1}mdb,cn=config
changetype: modify
delete: olcAccess
olcAccess: {6}
-
delete: olcAccess
olcAccess: {5}
-
delete: olcAccess
olcAccess: {4}
```

Listing 17.49 Löschen mehrerer ACLs

17.12.1 ACLs vor dem Einsatz testen

Um die ACLs zu testen, verwenden Sie das Kommando `slapacl`. Das Kommando `slapacl` wird, wie alle anderen `slap`-Kommandos auch, beim Kompilieren des OpenLDAP gebaut. Aus diesem Grund gelten hier auch die Standardeinstellungen für die Art und Weise, wie auf die Konfiguration zugegriffen wird. Der Standard bei allen Distributionen ist daher die dynamische Konfiguration. Für den Aufruf des Kommandos bedeutet das, dass Sie bei der statischen Konfiguration bei Verwendung der `slap`-Kommandos immer den Pfad zur Datei `slapd.conf` mit angeben müssen. In den Beispielen haben wir das immer getan. Beim Verwenden der dynamischen Konfiguration lassen Sie den Parameter `-f /etc/ldap/slapd.conf` einfach weg. In Listing 17.50 sehen Sie ein paar Beispiele dazu:

Beispiel 1:

```
slapacl -D "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net" \
        -b "cn=u2 prod,ou=users,ou=produktion,dc=example,dc=net"
authcDN: "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net"
entry: read(=rscxd)
children: read(=rscxd)
objectClass=posixAccount: read(=rscxd)
objectClass=inetOrgPerson: read(=rscxd)
objectClass=organizationalPerson: read(=rscxd)
objectClass=person: read(=rscxd)
loginShell=/bin/bash: read(=rscxd)
homeDirectory=/home/u2-prod: read(=rscxd)
uid=u2-prod: read(=rscxd)
cn=u2 Prod: read(=rscxd)
userPassword=****: none(=0)
uidNumber=11002: read(=rscxd)
```

```
gidNumber=11000: read(=rscxd)
sn=Prod: read(=rscxd)
givenName=u2: read(=rscxd)
structuralObjectClass=inetOrgPerson: read(=rscxd)
entryUUID=38803892-0a5d-103d-80c1-fb9fccda7760: read(=rscxd)
creatorsName=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth: read(=rscxd)
createTimestamp=20221207092815Z: read(=rscxd)
entryCSN=20221207092815.051335Z#000000#000#000000: read(=rscxd)
modifiersName=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth: read(=rscxd)
modifyTimestamp=20221207092815Z: read(=rscxd)
```

Beispiel 2:

```
slapacl -f /etc/ldap/slapd.conf \
-D "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net"
-b "ou=cn=u2 prod,ou=users,ou=produktion,dc=example,\
dc=net" userpassword
authcDN: "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net"
userPassword: none(=0)
```

Beispiel 3:

```
slapacl -f /etc/ldap/slapd.conf \
-D "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net" \
-b "cn=u2 prod,ou=users,ou=produktion,\
dc=example,dc=net" userpassword/write
authcDN: "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net"
write access to userPassword: DENIED
```

Listing 17.50 Beispiele für das Kommando »slapacl«

In der folgenden Liste finden Sie die Erklärungen zu den Beispielen:

► Beispiel 1

Hier werden die Rechte des Benutzers `-D cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net` an dem Objekt `-b cn=u2 prod,ou=users,ou=produktion,dc=example,dc=net` geprüft. Es werden alle Attribute mit den entsprechenden Rechten aufgelistet. Im Beispiel können Sie sehr gut sehen, dass der Benutzer an dem Passwort des anderen Benutzers keine Rechte hat. Führen Sie den Test erneut durch und verwenden dann das eigene Objekt des Benutzers werden Sie sehen, dass er sein eigenes Passwort ändern kann.

► Beispiel 2

Hier wird nur nach dem Passwort des Benutzers geschaut, und es werden alle Rechte angezeigt, die der Benutzer an dem Attribut selbst hat. So können Sie gezielt einzelne Attribute abfragen.

► Beispiel 3

Bei diesem Beispiel wird nur geprüft, ob der Benutzer das Schreibrecht an dem Attribut `userPassword` hat. Als Antwort kommt nur ein `DENIED`. Sollte das Recht vorhanden sein, würde die Antwort `ALLOWED` erscheinen.

Seien Sie aber vorsichtig mit dem Ergebnis! Schauen Sie sich das Ergebnis des ersten Tests noch einmal an. Dort finden Sie die Zeile `children: read(=rscxd)`. Das bedeutet, dass der *u1 prod* auf alle untergeordneten Objekte von *u2 prod* zugreifen darf. Das stimmt so aber nicht. Denn der Zugriff auf die OU `ou=adressen` des Objekts ist durch eine andere Regel verboten – siehe Listing 17.51:

```
slapacd -D "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net" \
        -b "ou=adressen,cn=u2 prod,ou=users,ou=produktion,dc=example,dc=net"
authcDN: "cn=u1 prod,ou=users,ou=produktion,dc=example,dc=net"
entry: none(=0)
children: none(=0)
ou=Adressen: none(=0)
objectClass=organizationalUnit: none(=0)
objectClass=top: none(=0)
structuralObjectClass=organizationalUnit: none(=0)
entryUUID=3893dbd6-0a5d-103d-80cd-fb9fccda7760: none(=0)
creatorsName=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth: none(=0)
createTimestamp=20221207092815Z: none(=0)
entryCSN=20221207092815.180029Z#000000#000#000000: none(=0)
modifiersName=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth: none(=0)
modifyTimestamp=20221207092815Z: none(=0)
```

Listing 17.51 Kein Zugriff auf die »ou=adressen«

Sollte der Zugriff auf ein untergeordnetes Objekt nicht klappen, schauen Sie immer in die ACLs, ob der Zugriff auf diese Objekte noch explizit durch eine eigene ACL eingeschränkt wird. Es ist immer eine gute Idee, jede neue ACL mit dem Kommando `slapacd` daraufhin zu überprüfen, ob auch das von Ihnen gewünschte Ergebnis mit der ACL erreicht wird.

17.13 Filter zur Suche im LDAP-Baum

Nachdem Sie den DIT mit Objekten ausgestattet haben, werfen wir an dieser Stelle einen Blick auf die Filter, die zur Suche im DIT verwendet werden können. Der Beispiel-DIT ist ja sehr überschaubar, aber wenn OpenLDAP in einem Unternehmen eingeführt wird, kann der DIT schon sehr viel größer werden. Dann kann es schon schwieriger sein, ein ganz bestimmtes Objekt zu finden, besonders dann, wenn Sie nicht genügend Informationen über das Objekt oder bestimmte Werte von Attributen haben. Deshalb ist es wichtig zu wissen, wie Sie mithilfe von Filtern die Suche einschränken können.

17.13.1 Die Fähigkeiten des LDAP-Servers testen

Mithilfe der Filter können Sie sich auch die Fähigkeiten des LDAP-Servers anschauen. Nutzen Sie dazu das Kommando aus Listing 17.52:

```
ldapsearch -x -b "" -s base + -LLL
dn:
structuralObjectClass: OpenLDAProotDSE
configContext: cn=config
monitorContext: cn=Monitor
namingContexts: dc=example,dc=net
supportedControl: 1.3.6.1.4.1.4203.1.9.1.1
supportedControl: 1.2.840.113556.1.4.619
supportedControl: 1.3.6.1.4.1.21008.108.63.1
supportedControl: 2.16.840.1.113730.3.4.18
supportedControl: 2.16.840.1.113730.3.4.2
supportedControl: 1.3.6.1.1.21.2
supportedControl: 1.3.6.1.4.1.4203.666.5.12
supportedControl: 1.3.6.1.4.1.4203.666.5.2
supportedControl: 1.3.6.1.4.1.4203.1.10.1
supportedControl: 1.2.840.113556.1.4.1340
supportedControl: 1.2.840.113556.1.4.805
supportedControl: 1.2.840.113556.1.4.1413
supportedControl: 1.3.6.1.1.22
supportedControl: 1.2.840.113556.1.4.1339
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.826.0.1.3344810.2.3
supportedControl: 1.3.6.1.1.13.2
supportedControl: 1.3.6.1.1.13.1
supportedControl: 1.3.6.1.1.12
supportedExtension: 1.3.6.1.4.1.1466.20037
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.4203.1.11.3
supportedExtension: 1.3.6.1.1.8
supportedExtension: 1.3.6.1.1.21.3
supportedExtension: 1.3.6.1.1.21.1
supportedFeatures: 1.3.6.1.1.14
supportedFeatures: 1.3.6.1.4.1.4203.1.5.1
supportedFeatures: 1.3.6.1.4.1.4203.1.5.2
supportedFeatures: 1.3.6.1.4.1.4203.1.5.3
supportedFeatures: 1.3.6.1.4.1.4203.1.5.4
supportedFeatures: 1.3.6.1.4.1.4203.1.5.5
supportedLDAPVersion: 3
supportedSASLMechanisms: SCRAM-SHA-512
```

```

supportedSASLMechanisms: SCRAM-SHA-384
supportedSASLMechanisms: SCRAM-SHA-256
supportedSASLMechanisms: SCRAM-SHA-224
supportedSASLMechanisms: SCRAM-SHA-1
supportedSASLMechanisms: GSS-SPNEGO
supportedSASLMechanisms: GSSAPI
supportedSASLMechanisms: DIGEST-MD5
supportedSASLMechanisms: OTP
supportedSASLMechanisms: CRAM-MD5
supportedSASLMechanisms: PLAIN
supportedSASLMechanisms: LOGIN
entryDN:
subschemaSubentry: cn=Subschema

```

Listing 17.52 Liste der Fähigkeiten des LDAP-Servers

Das Kommando `ldapsearch` wird hier nur auf der obersten Ebene ausgeführt, und durch das Plus bei den Kommandoparametern werden Ihnen zusätzliche Fähigkeiten und Funktionen des LDAP-Servers angezeigt. Neben den `NamingContexts` werden noch die Einträge `supportedControl`, `supportedExtension` und `supportedFeatures` mit den entsprechenden OIDs gekennzeichnet. Eine weitere wichtige Information liefern die `supportedSASLMechanisms`, die der LDAP-Server versteht. Die Bedeutung der einzelnen Informationen wird hier kurz erklärt:

- ▶ **supportedControl**
Über die `supportedControl`-Einträge kann das Standardverhalten des Servers gegenüber dem Client geändert werden.
- ▶ **supportedExtension**
Hier werden zusätzliche Erweiterungen von Funktionen beschrieben.
- ▶ **supportedFeatures**
Hiermit kann der Server dem Client mitteilen, ob eine bestimmte Funktion von ihm unterstützt wird.
- ▶ **supportedSASLMechanisms**
Hier werden alle Verschlüsselungsmechanismen aufgelistet, die der LDAP-Server bei der Verwendung von SASL unterstützt. Die Mechanismen müssen schon beim Kompilieren des LDAP-Servers mit angegeben werden.

17.13.2 Einfache Filter

In diesem Abschnitt gehen wir etwas näher auf die Suche nach bestimmten Attributen ein. Sie können die Suche auf einzelne Attribute (wie `uid=skania`) bis hin zu Objektklassen (wie `objectClass=inetOrgPerson`) beziehen. Auch ist es möglich, mit dem Stern als Jokerzeichen zu arbeiten, wobei der Stern auch nur Teile eines Suchstrings ersetzen kann.



Bei allen Beispielen kommt *ldaps* zum Einsatz, sodass der Parameter `-ZZ` beim Kommando `ldapsearch` nicht benötigt wird.

An den folgenden Beispielen soll Ihnen die Verwendung der Filter erklärt werden:

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W "(uid=*)"
```

Listing 17.53 Suche nach allen Objekten mit dem Attribut »uid«

Mit der Suche in Listing 17.53 werden alle Objekte angezeigt, bei denen das Attribut `uid` vorhanden ist und einen beliebigen Wert hat.

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W "(uid=s*)"
```

Listing 17.54 Suche nach allen Objekten, deren »uid« mit »s« beginnt

In Listing 17.54 muss die `uid` mit einem `s` beginnen, damit das Objekt angezeigt wird. Wenn Sie sich zu dem Attribut `uid` den entsprechenden Eintrag im Schema anschauen, werden Sie sehen, dass hier nicht zwischen Groß- und Kleinschreibung unterschieden wird. In Listing 17.55 wird jetzt nach einer ganz bestimmten `uid` gesucht. Angezeigt werden aber nur der `dn` und die Attribute `cn` und `sn`.

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W "(uid=skania)" cn sn
```

Listing 17.55 Anzeige von bestimmten Attributen bei einer Suche



Leerzeichen statt Komma

Achten Sie darauf, dass in der Liste der Attribute, die angezeigt werden sollen, diese mit einem Leerzeichen getrennt werden und nicht durch ein Komma wie bei den ACLs.

17.13.3 Filter mit logischen Verknüpfungen

Sie können die Filter auch durch logische UND- oder ODER-Verknüpfungen erweitern. Dadurch ergeben sich sehr viele Möglichkeiten, die Filter zu kombinieren. Neben der logischen Verknüpfung können Sie bestimmte Werte auch negieren. Anhand der folgenden Beispiele sollen Ihnen diese Möglichkeiten gezeigt werden. Für die ODER-Verknüpfung in Listing 17.56 wird das Pipe-Symbol verwendet. Das Symbol muss vor allen Attributen stehen, die verknüpft werden sollen. Sie können auch mehr als zwei Attribute miteinander verknüpfen. Jedes Attribut wird dann in eine eigene Klammer geschrieben:

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" \
  -W "(|(uid=s*)(uid=u*))" uid
```

Listing 17.56 ODER-Verknüpfung zweier Suchkriterien

Auch UND-Verknüpfungen sind möglich, wie Sie in Listing 17.57 sehen können. Als Zeichen für die UND-Verknüpfung verwenden Sie das Kaufmanns-Und (&). Sonst gelten die gleichen Regeln wie bei der ODER-Verknüpfung:

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W \
    "&(uid=s*)(loginShell=/bin/bash)" uid
```

Listing 17.57 UND-Verknüpfung zweier Suchkriterien

Auch eine Negation des Suchbegriffs ist möglich, wie Sie in Listing 17.58 sehen. Um eine Negation einzuleiten, setzen Sie vor das zu negierende Attribut ein Ausrufezeichen.

Im Beispiel aus Listing 17.58 darf die uid nicht mit dem Buchstaben c beginnen. Die Negation bezieht sich nur auf das eine Attribut und nicht auf die Kette der Attribute:

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W \
    "&(!(uid=c*))(loginShell=/bin/bash)"
```

Listing 17.58 Negation von Suchkriterien

17.13.4 Einschränkung der Suchtiefe

Oft möchten Sie vielleicht gar nicht den gesamten LDAP-Baum durchsuchen, sondern nur eine bestimmte Ebene oder nur einen Teilbaum. Auch dafür gibt es Möglichkeiten, die wir Ihnen hier an einigen Beispielen zeigen wollen:

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W \
    -b "ou=users,dc=example,dc=net" -s sub "&(!(uid=*)))(sn=*)"
```

Listing 17.59 Einschränkung der Suchtiefe

Durch die Angabe der Optionen `-b "ou=users,dc=example,dc=net"` und `-s sub`, die Sie in Listing 17.59 sehen, können Sie den Startpunkt der Suche und die Suchtiefe bestimmen. Neben den Attributen, die Sie beim Erstellen oder Ändern von Objekten angegeben haben, gibt es weitere interne Attribute, die zur Verwaltung der Objekte benötigt werden.

Hier werden unter anderem die Erstellungszeit und der Ersteller gespeichert. Auch bekommt jedes Objekt eine eindeutige ID, die an dieser Stelle verwaltet wird. Einige der Attribute werden auch für die Replikation des DIT benötigt.

In Listing 17.60 sehen Sie das Kommando, mit dem Sie sich die Systemattribute anzeigen lassen können:

```
ldapsearch -x -D "uid=ldap-admin,ou=users,dc=example,dc=net" -W "(uid=s*)" +
```

Listing 17.60 Anzeige der Systemattribute

17.14 Verwendung von Overlays

Bei den Overlays handelt es sich um Funktionserweiterungen des OpenLDAP-Servers. Diese Erweiterungen kann man sich wie eine Art Plug-ins vorstellen, die über eine Schnittstelle zur Verfügung gestellt werden. Über die Overlays können Sie neue Funktionen hinzufügen, ohne OpenLDAP selbst neu kompilieren zu müssen.

Bei einer Anfrage an den LDAP-Server wird die Anfrage vom LDAP-Frontend entgegengenommen, dort interpretiert und anschließend an das Backend zur Verarbeitung weitergeleitet. Wenn das Backend die Anfrage abgearbeitet hat, wird das Ergebnis an das Frontend weitergeleitet. Das Frontend leitet das Ergebnis an den Client weiter, der die Anfrage gestellt hat. Die Overlays können nun zwischen das Frontend und das Backend geschaltet werden. Das Overlay kann dann die Anfrage verändern oder zusätzlich Aktionen starten, bevor die Anfrage an das Backend weitergeleitet wird. Es ist auch möglich, dass die Ergebnisse des Backends durch ein Overlay verändert werden, bevor es zum Frontend geleitet wird. Zudem ist es möglich, mehrere Overlays hintereinanderschalten, wobei auf die Reihenfolge geachtet werden muss, da die Ergebnisse je nach Reihenfolge unterschiedlich ausfallen können.

Overlays können einerseits im globalen Teil der Konfiguration geladen werden. Dadurch sind die Overlays für alle Datenbanken gültig, die auf diesem LDAP-Server eingerichtet wurden. Wird das Overlay andererseits in einem Datenbankbereich der *slapd.conf* geladen, ist es nur für diese Datenbank aktiv. Die Möglichkeit, Overlays zu verwenden, besteht seit der OpenLDAP-Version 2.2. Mit der Zeit sind eine Menge Overlays bereitgestellt worden. Einige sind Bestandteil des OpenLDAP-Servers, andere wurden von Anwendern programmiert und zur Verfügung gestellt. Welche Overlays der OpenLDAP-Server der jeweiligen Distribution bereitstellt, können Sie mit dem Kommando `apropos slapo-` überprüfen. Die Funktionen der dort aufgelisteten Overlays können Sie in den entsprechenden Manpages nachlesen.

17.14.1 Overlays am Beispiel von »dynlist«

Wenn Sie die Objekte für die Beispiele der ACLs angelegt haben, wurden dabei auch die beiden Benutzer `cn=prod al` und `cn=verw al` angelegt. Bei diesen beiden soll es sich um die Abteilungsleiter der entsprechenden Abteilung handeln. Jetzt soll über eine Gruppe geregelt werden, dass alle Abteilungsleiter, auch die von zukünftigen Abteilungen, automatisch Mitglied der Gruppe `cn=abteilungsleiter,ou=group,dc=example,dc=net` werden sollen. Die Mitglieder der Gruppe sollen dann für alle Benutzer in ihrer Abteilung das Recht bekommen, die Passwörter zurücksetzen zu können.

Mit dem Overlay `dynlist` können Sie diese Aufgabe automatisch durchführen lassen. Bevor Sie das Overlay nutzen können, ist es erforderlich, dass Sie zuerst noch die Konfiguration des LDAP-Servers um das Schema `dyngroup` erweitern.



Wenn Sie unsere Basiskonfiguration bei der Einrichtung des OpenLDAP genutzt haben, ist das Schema bereits in der Konfiguration enthalten.

Dann kann das Overlay konfiguriert werden, und daraufhin können Sie die ersten dynamischen Gruppen anlegen. Alle drei Schritte werden wir Ihnen jetzt ausführlich beschreiben.

Einbinden des Schemas in die Konfiguration

Bei der dynamischen Konfiguration binden Sie das Schema über eine LDIF-Datei ein. Sie benötigen die Schemadatei im LDIF-Format. Da es sich bei der Schemadatei um eines der Schemata handelt, die mit OpenLDAP installiert wurde, ist diese Datei bereits vorhanden. Es handelt sich um die Datei `/opt/symas/etc/openldap/schema/dyngroup.ldif`. Wenn Sie jetzt, wie schon bei den ACLs, einfach versuchen, die LDIF-Datei mit dem Kommando `ldapmodify` einzuspielen, werden Sie die Fehlermeldung aus Listing 17.61 erhalten:

```
ldapmodify -Y EXTERNAL -H ldapi:/// -f /opt/symas/etc/openldap/schema/dyngroup.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
ldapmodify: modify operation type is missing at line 2, \
entry "cn=dyngroup,cn=schema,cn=config"
```

Listing 17.61 Fehler beim Einspielen der LDIF-Datei

Editieren Sie die LDIF-Datei des Schemas, und fügen Sie die fehlende Zeile aus Listing 17.62 an derselben Stelle in die Datei ein – dann können Sie die LDIF-Datei wie gewohnt in die Konfiguration einspielen:

```
dn: cn=dyngroup,cn=schema,cn=config
changetype: add
objectClass: olcSchemaConfig
```

Listing 17.62 Einfügen des »changetype« in die LDIF-Datei

Einrichtung des Overlay »dynlist«

Die nötigen Einträge für die Konfiguration spielen Sie wieder über eine LDIF-Datei ein. Die Konfiguration für das Overlay ist auch hier auf zwei Teile aufgeteilt: Einmal benötigen Sie das entsprechende Modul, das geladen werden muss, und dann die Konfiguration des Moduls. Das Schema wurde ja bereits eingebunden, und das finden Sie jetzt auch schon in Ihrer Konfiguration. In Listing 17.63 sehen Sie die LDIF-Datei für die Einrichtung des Overlays:

```
dn: cn=module0,cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: dynlist.la
```

```
dn: olcOverlay=dynlist,olcDatabase=2mdb,cn=config
changetype: add
objectClass: olcDynamicList
objectClass: olcOverlayConfig
olcOverlay: dynlist
olcDlAttrSet: groupOfURLs memberURL member
```

Listing 17.63 LDIF für die dynamische Konfiguration

Sie sehen in diesem Beispiel, dass zuerst das Modul in die Konfiguration eingetragen wird und dass im zweiten Schritt das Modul konfiguriert wird. Da dieses das erste zusätzliche Modul ist, schauen Sie sich hier auch mal die Änderungen in der Konfiguration an. Am Anfang der Konfiguration wird das Modul einfach zur Liste der zu ladenden Module hinzugefügt. Die Konfiguration für das Overlay selbst finden Sie ganz am Ende der Konfiguration. Die Konfiguration wird hier direkt in die Objektdatenbank eingetragen, das können Sie am `dn`: erkennen. Der `dn`: erzeugt einen eigenen Abschnitt für das Overlay in der Datenbank mit der Nummer 2. Bei dieser Datenbank handelt es sich um die Objektdatenbank. Das Einspielen des neuen Moduls und die Konfiguration sehen Sie in Listing 17.64:

```
root@ldap01:~# ldapmodify -Y EXTERNAL -H ldapi:/// -f dynlist-konfig.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=module0,cn=config"
```

```
adding new entry "olcOverlay=dynlist,olcDatabase=2mdb,cn=config"
```

Listing 17.64 Einspielen des Moduls

Für die Konfiguration des Overlays wird ein eigener Bereich in der Konfiguration erzeugt, der über eine Nummer der Datenbank, zugeordnet wird. Haben Sie mehrere Datenbanken auf Ihrem Server eingerichtet, erstellen Sie für jede Datenbank eine eigene Zuordnung und Konfiguration für das Overlay. Diese Vorgehensweise gilt für alle Overlays, die Sie direkt in Datenbanken eintragen.

Achten Sie auf die Datenbanknummer

Wenn Sie nicht unsere Beispielkonfiguration nutzen, achten Sie unbedingt auf die korrekte Nummer der Datenbank. Sie können Datenbanken im OpenLDAP in unterschiedlichen Reihenfolgen einrichten, die Nummern der Datenbanken können daher unterschiedlich sein.

Erstellen einer dynamischen Gruppe

Erst jetzt können Sie die LDIF-Datei für die Gruppe vom Typ `groupOfURLs` aus Listing 17.65 erstellen und in den LDAP-Baum einspielen:

```
dn: cn=abteilungsleiter,ou=groups,dc=example,dc=net
objectClass: groupOfURLs
cn: abteilungsleiter
memberURL: ldap:///dc=example,dc=net??sub?(employeeType=Abteilungsleiter)
```

Listing 17.65 LDIF-Datei für eine dynamische Gruppe

Die Zeile `memberURL` soll jetzt extra erklärt werden:

- ▶ Verweis auf den Suffix `cd=example,dc=net` auf dem lokalen Server
- ▶ soll durch den gesamten Baum `sub`
- ▶ nach allen Objekten `??` gesucht werden,
- ▶ die ein Attribut `employeeType`
- ▶ mit dem Wert `Abteilungsleiter` besitzen.

Diese Objekte sollen dann mit ihrem vollständigen DN in die Liste der Gruppen aufgenommen werden. Die Zeile darf keine Leerzeichen enthalten, deshalb auch die zwei aufeinander folgenden Fragezeichen. Sie hätten bei der dynamischen Gruppe auch noch die Möglichkeit, die Mitglieder nur mit ihrem Namen ohne den gesamten DN zu verwenden (statt `cn=user,ou=users,dc=example,dc=net` nur `user`). Dann würden Sie zwischen die zwei Fragezeichen noch den Objekttyp (im Falle eines Benutzers `uid`) eintragen. Anschließend können Sie die Gruppe aber nicht mehr dazu nutzen, Rechte im LDAP zu vergeben – was hier aber passieren soll. Wenn Sie die Objekte für die beiden Abteilungen eingespielt haben, dann besitzen die beiden Abteilungsleiter der Abteilungen bereits das entsprechende Attribut. Sollten Sie noch keine Benutzer mit dem entsprechenden Attribut angelegt haben, können Sie das Attribut mithilfe der LDIF-Datei aus Listing 17.66 für einen Benutzer setzen:

```
dn: cn=prod al,ou=users,ou=produktion,dc=example,dc=net
changeType: modify
add: employeeType
employeeType: Abteilungsleiter
```

Listing 17.66 Setzen des Attributs für die dynamische Gruppe

Hat ein Benutzer bereits einen Eintrag `employeeType`, können Sie diesen auch überschreiben. Dazu ist es lediglich notwendig, dass Sie `add: employeeType` durch `replace: employeeType` ersetzen. Wollen Sie das Attribut ganz löschen, verwenden Sie die LDIF-Datei aus Listing 17.67:

```
dn: cn=prod al,ou=users,ou=produktion,dc=example,dc=net
changeType: modify
delete: employeeType
```

Listing 17.67 LDIF-Datei zum Entfernen des Attributs »employeeType«



Verwendung von »replace«

Wenn Sie nicht erst kontrollieren wollen, ob das Attribut gesetzt ist oder nicht, können Sie zum Hinzufügen auch immer `replace` verwenden. Denn wenn das Attribut schon gesetzt ist, wird es ersetzt, und wenn das Attribut nicht gesetzt ist, wird es dadurch gesetzt. Verwenden Sie `add` und ist das Attribut bereits vorhanden, kommt es zu einer entsprechenden Fehlermeldung.

Besonderheit von dynamischen Gruppen

Das Overlay für die dynamischen Gruppen sorgt dafür, dass bei jedem Auflisten der Gruppe mittels `ldapsearch` alle Benutzer durchsucht werden und dann die Mitglieder dynamisch zugewiesen werden. Die Gruppenmitgliedschaften werden nicht in der Datenbank eingetragen. Das können Sie sehr einfach kontrollieren. Lassen Sie sich die Gruppe einmal mit `ldapsearch` anzeigen, und nutzen Sie einmal `slapcat` für die Auflistung aller Objekte. Dann sehen Sie, dass beim Ergebnis von `ldapsearch` die Mitglieder angezeigt werden, bei der Verwendung von `slapcat` aber nicht.

Ein weiterer Punkt soll hier nicht unerwähnt bleiben: Ab der OpenLDAP-Version 2.5 ist das Overlay `memberof` als `deprecated` gekennzeichnet. Auf die Dauer wird das Overlay nicht mehr zur Verfügung stehen. Das Overlay `dynlist` wird die Funktion übernehmen. Wenn Sie noch das Overlay `memberof` nutzen, sollten Sie sich ein Migrationsstrategie überlegen. Mehr zu der Funktion finden Sie in der Manpage `slapo-dynlist`.

17.14.2 Weitere Overlays

Neben dem genannten Overlay gibt es eine Menge weiterer Overlays, die Sie in die Konfiguration Ihres LDAP-Servers einbinden können. Alle können hier natürlich nicht erklärt werden. Besonders interessant sind aber folgende Overlays:

► **dds (dynamic directory services)**

Mit diesem Overlay können Sie Objekte erzeugen, die nur eine beschränkte Lebensdauer haben. So wäre es möglich, dass Sie eine OU anlegen, in der Sie solche Benutzer anlegen, die nur für eine bestimmte Zeit im Unternehmen sind. Nach dem Ablauf der von Ihnen vorgegebenen Zeit würde das Objekt automatisch gelöscht.

► **refint (referenzielle Integrität)**

Das Overlay `refint` sorgt dafür, dass bei einer Änderung oder bei einer Löschung eines Objekts im gesamten DIT die Einträge zu diesem Objekt geändert oder entfernt werden.

Wenn Sie dieses Overlay aktivieren und einen Benutzer aus einer Gruppe entfernen oder seine UID ändern, wird der gesamte DIT durchsucht und alle weiteren Vorkommen dieses Namens werden auch gelöscht oder geändert.

► **valsort (value sort)**

Wenn Sie sich die Liste der Mitglieder einer Gruppe anzeigen lassen, wird Ihnen die Liste immer in der Reihenfolge angezeigt, in der Sie die Benutzer zu der Gruppe hinzugefügt haben. Mit dem Overlay `valsort` lässt sich diese Liste sortiert anzeigen. Das Sortieren findet auf dem Server statt, somit müssen Sie keine Änderungen an dem Client vornehmen.

► **chain**

Mit dem Overlay `chain` können Anfragen an einen LDAP-Server in einem anderen Referral weitergeleitet werden. Dadurch können Sie auch nach Objekten in einem anderen Referral suchen. Eine andere Möglichkeit, die Sie wahrscheinlich häufiger nutzen werden, ist die Option, schreibende Anfragen von einem Consumer an den Provider weiterzuleiten. So haben Sie die Möglichkeit, dass Ihre Clients alle einen Consumer als LDAP-Server in der `ldap.conf` eingetragen haben, aber trotzdem können die Benutzer ihr Passwort ändern. Die Änderung wird dann vom Consumer an den Provider weitergeleitet, dort in die Datenbank eingetragen und anschließend an alle Consumer repliziert. Der Consumer selbst wird nie schreibend auf die Datenbank zugreifen.

Neue Overlays ab OpenLDAP 2.5

Mit dem Erscheinen von OpenLDAP 2.5 gibt es neben der Änderung zu `dynlist` auch noch einige neue Overlays, die wir Ihnen hier vorstellen wollen:

► **autoca**

Mithilfe des Overlays `autoca` können Sie automatisch Zertifikate für Benutzer und Server erzeugen lassen. Neben der automatisch bei der Installation eingerichteten CA können Sie auch eigene CA-Zertifikate im LDAP hinterlegen.

► **otp**

Wenn Sie das Overlay `otp` einsetzen, können Sie für Ihre Benutzer eine *Zwei-Faktor-Authentifizierung (2FA)* einrichten. Das Overlay unterstützt *Time-based One-time Passwords* und *HMAC-based One-time Passwords*.

► **deref**

Mit dem Overlay `deref` können Sie die Suche nach Gruppenmitgliedern vereinfachen. Nach der Einrichtung des Overlays können Sie dann die Suche mit dem Kommando `ldapsearch -x -E 'deref=member:uid'` durchführen.

Neben diesen Overlays gibt es noch weitere neue Overlays, die aber so speziell sind, dass sie hier den Rahmen sprengen würden. Leider können wir aus Platzgründen hier im Buch auch nicht auf die Konfiguration aller neuen Overlays eingehen. Aber alle Manpages zu den Overlays sind sehr ausführlich und mit Beispielen versehen.

17.15 Replikation des DIT

In jedem Netzwerk kommt es früher oder später einmal zum Ausfall von Hardware oder Serverdiensten. Damit Ihr Netzwerk nicht ohne einen Anmeldeserver ist, falls der LDAP-Server einmal ausfällt, sollten Sie sich frühzeitig über die Ausfallsicherheit des LDAP-Servers Gedanken machen. Die einfachste Möglichkeit, um beim Ausfall eines LDAP-Servers nicht den gesamten LDAP-Dienst im Netzwerk zu verlieren, ist eine Replikation des gesamten DIT auf einen oder mehrere zusätzliche Server. Durch die Replikation ist es auch möglich, an verschiedenen Standorten Ihres Unternehmens je einen LDAP-Server zu installieren, und die Benutzer melden sich immer im lokalen Netzwerk am LDAP-Server an. In diesem Abschnitt soll jetzt die Replikation des LDAP-Servers realisiert werden. Mit der OpenLDAP-Version 2.5 hat sich auch im Bereich der Replikation einiges getan. So ist es jetzt auch möglich, die dynamische Konfiguration zwischen Providern zu replizieren. Ja, die Möglichkeit bestand vorher auch schon, sie funktionierte aber nicht so gut, dass sie für den Produktiveinsatz tauglich gewesen wäre. Weiterhin können Sie zwei Provider in einer *Standby-Provider-Replication* betreiben, aber wegen der verbesserten Replikationsmöglichkeiten ist es nun sinnvoller, die Provider mit einer *Multi Provider Replication* zu betreiben.

Die Replikation wird beim *syncrepl* zwischen dem *Provider* und dem *Consumer* oder zwischen mehreren *Providern* durchgeführt. Bei der Replikation via *syncrepl* wird kein eigenständiger Daemon genutzt, sondern es wird lediglich ein zusätzlicher Thread auf dem Consumer erstellt und verwaltet. Beim ersten Kontakt zwischen Consumer und Provider findet eine vollständige Replikation zwischen Consumer und Provider statt. Im Anschluss daran kann der Consumer regelmäßig die Updates vom Provider ziehen oder zu bestimmten Zeiten eine Verbindung zum Provider herstellen und den DIT dann wieder abgleichen. Die Synchronisation wird beim *syncrepl* über zwei interne Attribute gesteuert, die jedes Objekt besitzt. Dabei handelt es sich um die Attribute *entryUUID* und *entryCSN*. Zusätzlich zu den beiden Attributen wird zwischen dem Provider und dem Consumer noch ein *Session Cookie* ausgetauscht. Die wesentlichen Vorteile von *syncrepl* sind:

- ▶ *syncrepl* kennt eine aktive und eine passive Replikation. Sollte einmal die Verbindung zwischen Provider und Consumer unterbrochen sein, wird der Consumer automatisch die Verbindung zum Provider wiederherstellen. Der passive Modus (in Bezug auf den Consumer) nennt sich *refreshAndPersist*. Dabei bleibt nach dem ersten Abgleich zwischen Consumer und Provider die Verbindung dauerhaft bestehen und alle Änderungen werden immer vom Provider automatisch und sofort an die Consumer verbreitet. So bleiben die Consumer immer auf dem aktuellen Stand.

Der aktive Modus nennt sich *refreshOnly*. In diesem Modus fragt der Consumer in regelmäßigen, in der Konfiguration festgelegten Zeitintervallen beim Provider nach Änderungen. Sind Änderungen auf dem Provider vorhanden, werden diese abgeglichen. Nachdem alle Änderungen übertragen wurden, sendet der Provider ein *Synchronisation Cookie* an

den Consumer. Das Cookie enthält die aktuelle *contextCSN*, einen Zeitstempel der letzten Änderung auf dem Provider und die eindeutige ID des Providers. Bei der nächsten Aktualisierung prüft der Provider das Cookie vom Consumer und weiß somit, auf welchem Stand sich der Consumer befindet.

- ▶ Da der *syncrepl* automatisch synchronisiert, kann die Datenbank des Consumers beim ersten Abgleich in jedem beliebigen Zustand sein, egal ob die Datenbank noch leer oder schon gefüllt ist.
- ▶ Es wird kein zusätzlicher Daemon benötigt.
- ▶ Es können auch nur bestimmte Teile oder auch nur bestimmte Attribute eines Objekts repliziert werden.
- ▶ Eine weitere Möglichkeit ist *delta-syncrepl*. Beim *delta-syncrepl* werden alle Änderungen auf dem Provider in einer eigenen Log-Datenbank abgelegt – aber nur die Attribute eines Objekts, die auch wirklich verändert wurden. In der Datenbank werden alle Änderungen für eine bestimmte Zeit gesammelt. Wenn im laufenden Betrieb der Consumer nach den Änderungen fragt, werden nur die entsprechenden Attribute der Objekte übertragen und nicht das ganze Objekt. Nur beim ersten Füllen der Datenbank oder wenn der Consumer die Verbindung zum Provider zu lange verloren hat, wird wieder die normale Synchronisation durchgeführt. Dieses Verfahren kann, besonders in sehr großen DITs, die Netzlast durch Replikationen reduzieren.

Da die Replikation sehr zeitabhängig ist, sollte die Systemzeit des Consumers und des Providers absolut gleich sein, damit es nicht zu Fehlern bei der Replikation kommt.



17.15.1 Vorbereitungen für die Replikation

Da wir hier eine Multiprovider-Replikation einrichten wollen, werden wir ausschließlich auf die dynamische Konfiguration eingehen. Auf den Providern wird als Erstes für die Datenbank, die repliziert werden soll, das Overlay *syncprov* konfiguriert. Wenn Sie die Multiprovider-Replikation einrichten wollen, ist jeder Provider immer auch Consumer; daher ist es notwendig, dass Sie das Overlay auf allen Providern einrichten. Bei der Konfiguration des Overlays *syncprov* werden Sie zwei Parameter konfigurieren: den *olcSpcheckpoint* und das *olcSpessionlog*.

Da die Replikation auf den beiden Parametern *entryCSN* und *entryUUID* beruht, ist es notwendig, beide Attribute mit Indexeinträgen zu versehen. Hier folgt jetzt eine Übersicht über die einzelnen Parameter:

- ▶ `index entryCSN,entryUUID eq`
Wenn ein Consumer auf den Provider zugreift, verwendet er die internen Attribute *entryCSN* und *entryUUID* aller Objekte, um Änderungen festzustellen. Durch diese Zeile werden aus den Attributen Indexdateien erstellt, die dann für die Suche verwendet werden. Da-

durch wird die Performance der Synchronisation erhöht. Diese Indexeinträge benötigen Sie auch auf allen Consumern.

- ▶ `overlay syncprov`
Durch diesen Eintrag wird die Synchronisation auf dem Provider geladen und somit gestartet.
- ▶ `olcSpcheckpoint 100 10`
Mit den Checkpoints werden die Schreiboperationen geprüft. Mit diesem Parameter wird festgelegt, wie häufig ein neuer Checkpoint festgelegt wird. Im Beispiel findet das nach 100 Änderungen oder nach 10 Minuten statt.
- ▶ `olcSpSessionlog 200`
Dieser Eintrag bestimmt die maximale Anzahl an Änderungen für ein Session-Log. Das Session-Log wird nicht auf der Festplatte abgelegt, sondern im Arbeitsspeicher. Die Größe des Session-Logs sollte im mindestens die maximale Anzahl an Objekten im DIT entsprechen.

Legen Sie ein *simpleSecurityObject* als Replikationsbenutzer an. Dieses Objekt wird später für den Zugriff vom Consumer auf den Provider verwendet.



Natürlich können Sie das auch direkt mit dem Admin Ihres DIT durchführen, aber sicherer ist die Variante über einen speziellen Benutzer. Dieser Benutzer hat ein eigenes Passwort und bekommt keine Schreibrechte auf dem Provider.

Das Objekt können Sie entweder über ein grafisches Werkzeug oder über eine LDIF-Datei erstellen. Listing 17.68 zeigt die LDIF-Datei für die Erzeugung des Objekts:

```
dn: uid=repl-user,ou=users,dc=example,dc=net
uid: repl-user
objectClass: account
objectClass: simpleSecurityObject
userPassword: {ARGON2}...
```

Listing 17.68 LDIF-Datei für einen Replikationsbenutzer

17.15.2 Einrichtung der Replikation

In diesem Abschnitt geht es erst einmal um die einfache Replikation zwischen einem Provider und einem Consumer. Im Anschluss soll dann ein zweiter Provider für die Ausfallsicherheit der beschreibbaren Instanz sorgen.

Einrichtung des Providers

Beginnen wollen wir die Konfiguration mit dem Anlegen der ACLs und des Limits für den Replikationsbenutzer. Dazu sehen Sie in Listing 17.69 den Inhalt der benötigten LDIF-Datei:


```

dn: olcDatabase={2}mdb,cn=config
changetype: modify
delete: olcAccess
olcAccess: {0}
-
add: olcAccess
olcAccess: {0}to *
    by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
    by dn.exact="uid=sss-d-user,ou=users,dc=example,dc=net" read
    by dn.exact="uid=repl-user,ou=users,dc=example,dc=net" read
    by dn.exact="uid=ldap-admin,ou=users,dc=example,dc=net" write
    by * break
-
add: olcLimits
olcLimits: {0} dn.exact="uid=repl-user,ou=users,dc=example,dc=net"
    time=unlimited
    size=unlimited

```

Listing 17.69 ACL und Limit für Replikationsbenutzer

Noch einmal der Hinweis: Achten Sie auf die Nummerierung der ACLs. Das Beispiel geht von den ACLs und Objekten aus, die wir am Anfang des Kapitels verwendet haben. Wenn Sie unsere Grundkonfiguration nutzen, sind diese Änderungen bereits in der Konfiguration eingetragen.



17

Nachdem Sie die ACLs und das Limit gesetzt haben, wird im nächsten Schritt das Modul für das Overlay syncprov geladen und in derselben LDIF-Datei gleich konfiguriert (Listing 17.70):

```

dn: cn=module{0},cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: syncprov.la

dn: olcOverlay=syncprov,olcDatabase={2}mdb,cn=config
changetype: add
objectClass: olcOverlayConfig
objectClass: olcSyncProvConfig
olcOverlay: syncprov
olcSpSessionlog: 300
olcSpCheckPoint: 100 10

```

Listing 17.70 Laden und Konfigurieren des Overlays »syncprov«

Jetzt fehlt nur noch ein Schritt: das Eintragen der Indizes für die beiden Attribute entryUUID und entryCSN. Die dafür benötigte LDIF-Datei sehen Sie in Listing 17.71:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
add: olcDbIndex
olcDbIndex: entryUUID,entryCSN eq
```

Listing 17.71 Indexeinträge für die Replikation

Selbstverständlich hätten Sie auch alle Änderungen in eine LDIF-Datei schreiben können, aber einige der Anpassungen benötigen Sie auch noch auf dem Consumer. Da wir die Änderungen thematisch auf einzelne LDIF-Dateien verteilt haben, ist es dann möglich, die LDIF-Dateien auch für die Consumer zu nutzen. Das werden wir Ihnen im folgenden Abschnitt zeigen. Jetzt ist der Provider konfiguriert und Sie können den ersten dynamisch konfigurierten Consumer einrichten.

Einrichtung des Consumers

Nachdem Sie die Symas-Pakete installiert haben, sorgen Sie als Erstes dafür, dass Sie eine eigene Grundkonfiguration auf dem Consumer erstellen. Die Vorgehensweise ist dabei identisch mit der auf dem Provider.

- ▶ Passen Sie die Datei `/etc/default/symas-openldapserver` für den Start über die dynamische Konfiguration an.
- ▶ Spielen Sie die Grundkonfiguration des LDAP-Servers mit dem Kommando `slapadd` wie schon beim Provider ein. Nehmen Sie auch hierfür wieder die Datei `config.ldif` aus den von uns bereitgestellten Dateien. Die Grundkonfiguration aller LDAP-Server soll später immer identisch sein.
- ▶ Sorgen Sie dafür, dass der Benutzer `openldap` und die Gruppe `openldap` Besitzer des Verzeichnisses `/opt/symas/etc/openldap/slapd.d` und aller Unterverzeichnisse sind. Vergeben Sie für Benutzer und Gruppe Lese- und Schreibrecht.
- ▶ Starten Sie den `symas-openldap-server` neu.
- ▶ Passen Sie die Datei `/opt/symas/etc/openldap/ldap.conf` an den Consumer an. Verwenden Sie hier im Moment noch `ldap://` als Protokoll.

Jetzt ist es notwendig, die Konfiguration auf den Stand des Providers zu bringen. In den nächsten Schritten werden wir aus diesem Grund erst die Schritte noch einmal wiederholen, die wir auf dem Provider am Anfang durchgeführt haben.

Setzen des Passworts für rootdn

Damit der `rootdn` ein Passwort bekommt, nutzen Sie die LDIF-Datei aus Listing 17.72:

```
dn: olcDatabase={0}config,cn=config
changetype: modify
add: olcRootPW
olcRootPW: {ARGON2}....
```

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
replace: olcRootPW
olcRootPW: {ARGON2}....
```

Listing 17.72 Passwort für »rootdn« festlegen

Wenn der Server von einem anderen Administrator als dem für den Provider verwaltet werden soll, dann können Sie hier auch ein anderes Passwort als auf dem Provider eintragen.

Eintragen der TLS-Einstellungen

Sorgen Sie dafür, dass die Zertifikatsdateien vorhanden sind und dass die Gruppe *openldap* das Leserecht an den Dateien besitzt. Auch das Rootzertifikat Ihrer CA kopieren Sie auf den Consumer. Passen Sie anschließend die LDIF-Datei für den Consumer so wie in Listing 17.73 an, und spielen Sie die Änderung ein:

```
dn: cn=config
changetype: modify
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/zertifikate/consumer-dyn-cert.pem
-
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/zertifikate/consumer-dyn-key.pem
-
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/zertifikate/demoCA/cacert.pem
```

Listing 17.73 LDIF-Datei für die TLS-Einstellungen

Testen Sie anschließend, ob Sie mit `ldapsearch -x -ZZ -LLL` eine Antwort vom LDAP-Server bekommen. Da die Objektdatenbank noch nicht repliziert wurde, bekommen Sie an dieser Stelle nur den Fehler `32 Object not found` zurück. Das Kommando soll nur testen, ob die Zertifikate alle nutzbar sind. Jetzt können Sie auch in der Datei *ldap.conf* das Protokoll auf *ldaps://* umstellen. Erst dann können Sie mit den nächsten Schritten fortfahren.

Anpassen der ACLs

Damit Sie jetzt nicht alle Schritte, mit denen wir die Vorgehensweise auf dem Provider erklärt haben, einzeln durchführen müssen, finden Sie in Listing 17.74 eine LDIF-Datei, die alle ACLs

identisch zum Provider setzt. Haben Sie bereits zusätzliche Einstellungen auf dem Provider vorgenommen, passen Sie die Datei entsprechend an:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
delete: olcAccess
olcAccess: {3}
-
delete: olcAccess
olcAccess: {2}
-
delete: olcAccess
olcAccess: {1}
-
delete: olcAccess
olcAccess: {0}
-
add: olcAccess
olcAccess: {0} to *
  by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth manage
  by dn.exact=uid=ldap-admin,ou=users,dc=example,dc=net write
  by dn.exact=uid=sssd-user,ou=users,dc=example,dc=net read
  by * break
-
add: olcAccess
olcAccess: {1}to dn.exact="dc=example,dc=net"
  by * read
-
add: olcAccess
olcAccess: {2} to attrs=userPassword
  by anonymous auth
  by group/groupOfNames/member=cn=pw-set,ou=groups,dc=example,dc=net write
  by self write
  by * none
-
add: olcAccess
olcAccess: {3}to attrs=shadowLastChange
  by self write by group/groupOfNames/member=cn=pw-set,\
  ou=groups,dc=example,dc=net write
  by * read
-
add: olcAccess
olcAccess: {4}to dn.regex=ou=adressen,cn=(.+),ou=users,\
  ou=(.+),dc=example,dc=net$
```

```

by dn.regex=^cn=$1,ou=users,ou=$2,dc=example,dc=net$ write
by * none
-
add: olcAccess
olcAccess: {5}to dn.sub=ou=verwaltung,dc=example,dc=net
  by dn.children=ou=users,ou=verwaltung,dc=example,dc=net read
  by * none
-
add: olcAccess
olcAccess: {6}to dn.sub=ou=produktion,dc=example,dc=net
  by dn.children=ou=users,ou=produktion,dc=example,dc=net read
  by * none
-
delete: olcLimits
olcLimits: {0}

```

Listing 17.74 Alle ACLs für den Consumer

Einrichtung des Overlays »dynlist«

Auch der Consumer benötigt die Overlays. Da wir im Beispiel des Providers das Overlay dynlist konfiguriert haben, benötigt auch der Consumer diese Konfiguration, da ansonsten die dynamischen Gruppen auf dem Consumer nicht ausgewertet werden können. Das benötigte Schema dyngroup wurde bereits mit der Grundkonfiguration zur Konfiguration hinzugefügt. Fehlt nur noch das Laden des Moduls und die Konfiguration des Moduls in der Objektdatenbank. Für das Laden des Moduls und die anschließende Konfiguration sehen Sie in Listing 17.75 die komplette LDIF-Datei:

```

root@ldap03:~# cat dynlist-conf.ldif
dn: cn=module{0},cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: dynlist.la

dn: olcOverlay=dynlist,olcDatabase={2}mdb,cn=config
changetype: add
objectClass: olcDynamicList
objectClass: olcOverlayConfig
olcOverlay: dynlist
olcDlAttrSet: groupOfURLs memberURL member

```

Listing 17.75 LDIF-Datei für das Overlay »dynlist«

Einrichtung der Replikation auf dem Consumer

Jetzt haben Sie die Konfiguration des Consumers so weit abgeschlossen, dass nur noch die Einrichtung der Replikation fehlt. Dazu wird die Objektdatenbank um die `syncrepl`-Direktive erweitert. Hier folgt jetzt nur noch die Einrichtung mittels der in Listing 17.76 gezeigten LDIF-Datei:

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
add:olcSyncrepl
olcSyncrepl: rid=001
    provider=ldaps://ldap01.example.net
    type=refreshAndPersist
    retry="10 5 400 5"
    searchbase="dc=example,dc=net"
    filter="(objectClass=*)"
    scope=sub
    schemachecking=off
    bindmethod=simple
    binddn="uid=repl-user,ou=users,dc=example,dc=net"
    credentials=geheim
```

Listing 17.76 LDIF-Datei für die »syncrepl«-Direktive

Ohne den *slapd* neu starten zu müssen, wird jetzt sofort die Replikation gestartet, und Sie können auf alle Objekte mit den entsprechenden Berechtigungen zugreifen. Die Parameter haben folgende Bedeutung:

- ▶ `syncrepl rid=001`
Das ist der Identifier für die Replikation. Ein Consumer kann die Informationen von mehreren Providern abholen, deshalb muss jeder Synchronisationseintrag eine eigene Nummer haben. Die dazugehörigen Einträge müssen alle eingerückt geschrieben werden, sonst kommt es beim Starten des LDAP-Servers zu Fehlermeldungen.
- ▶ `provider=ldaps://ldap01.example.net`
Das ist die URL des Providers, von dem alle Änderungen empfangen werden.
- ▶ `type=refreshAndPersist`
Dieser Parameter legt fest, wie der Provider und der Consumer die Daten aktuell halten. Der Typ `refreshAndPersist` erstellt eine dauerhafte Verbindung zwischen Provider und Consumer. Dadurch werden Änderungen schneller bekannt gegeben.

Wenn die beiden Server sich aber über eine langsame Leitung synchronisieren müssen, ist sicher der Typ `refreshOnly` besser angebracht: Hierbei initiiert der Consumer in regelmäßigen Abständen die Synchronisation mit dem Provider. Die Zeiten werden

über den Eintrag `interval` festgelegt. Der Eintrag kann zum Beispiel so aussehen: `interval=01:00:00:00`. Damit würde der Consumer den Provider einmal am Tag kontaktieren.

▶ `retry=10 5 400 5`

Dieser Eintrag legt fest, wie häufig der Consumer versucht, den Provider zu erreichen, wenn eine Synchronisation fehlschlägt. Die vier Werte haben die folgende Bedeutung:

1. Intervall alle 10 Sekunden
2. Anzahl der Wiederholungen: 5
3. Start der zweiten Intervallsequenz nach 400 Sekunden
4. Anzahl der Wiederholungen: 5. Wenn Sie nur ein `+` anstelle der 5 verwenden, wird der Consumer immer wieder versuchen den Provider zu erreichen (ohne Begrenzung).

▶ `searchbase=dc=example,dc=net`

Bei diesem Kontext startet die Suche nach Änderungen. Wenn Sie nicht den gesamten DIT replizieren wollen, können Sie an dieser Stelle auch Teilbäume angeben.

▶ `filter=(objectClass=*)`

Das sind Filter für die Objekte, die synchronisiert werden sollen. In diesem Fall werden alle Objekte repliziert, die eine Objektklasse besitzen. Da in unserem Beispiel alle Objekte eine Objektklasse besitzen, werden alle Objekte repliziert.

Einer der Vorteile von *syncrepl* ist, dass nicht alle Attribute repliziert werden müssen. Über die Verwendung von Filtern können Sie gezielt bestimmte Attribute auswählen, die repliziert werden sollen. Alle Filterkombinationen aus Abschnitt 17.13, »Filter zur Suche im LDAP-Baum«, können Sie an dieser Stelle verwenden.

▶ `scope=sub`

Der Eintrag `scope` legt die Suchtiefe fest. Hier soll der gesamte DIT durchsucht werden.

▶ `schemachecking=off`

Es erfolgt keine Prüfung der Schemata. Das ist immer dann besonders wichtig, wenn nur bestimmte Attribute repliziert werden sollen.

▶ `bindmethod=simple`

Wenn Sie `sasl` nicht eingerichtet haben, muss die Bind-Methode unbedingt `simple` sein.

▶ `binddn=uid=repl-user,ou=users,dc=example,dc=net`

Hier geben Sie den Benutzer an, der auf dem Provider verwendet werden soll, um die Einträge zu lesen. Achten Sie darauf, dass der Benutzer alle Attribute an den Objekten lesen kann, die repliziert werden sollen.

▶ `credentials=geheim`

Dies ist das Passwort für den Benutzer. Bei der Verwendung von *simple-bind* können Sie hier kein verschlüsseltes Passwort eintragen. Das zeigt auch noch einmal, warum Sie einen eigenen Benutzer für die Replikation erstellen sollten und warum die Kommunikation über TLS abgesichert werden sollte.

Damit ist auch die Replikation abgeschlossen. Wollen Sie auf dem Consumer den Benutzern im LDAP die Möglichkeit geben, sich am LDAP-Server anzumelden, ist es erforderlich, dass Sie den LDAP-Client *sssd* ebenfalls auf dem Consumer installieren und konfigurieren.

17.15.3 Einrichtung einer Multiprovider-Replikation

Bis zu diesem Zeitpunkt haben Sie nur einen Provider, aber eventuell schon mehrere Consumer eingerichtet. Solange Ihre Clients nur lesende Zugriffe auf den LDAP ausführen, ist der Ausfall des Providers kein Problem. Erst wenn Daten in der Objektdatenbank geändert oder neu erstellt werden sollen, wird ein ausgefallener Provider zum Problem. Um nun den Provider ausfallsicher machen zu können, ist es notwendig, dass alle Änderungen, die auf einem Provider erstellt werden, sofort auf einen oder mehrere zusätzliche Provider geschrieben werden. Dabei muss es unerheblich sein, auf welchem Ihrer Provider die Änderung durchgeführt wird. Auch ist es sinnvoll, dass alle Provider eine identische Konfiguration besitzen. Nur mit einer identischen Konfiguration können Sie Provider einfach und schnell austauschen.

Das Einrichten einer Multiprovider-Replikation ist dann sinnvoll, wenn Sie die dynamische Konfiguration einsetzen. Mit dem Erscheinen von OpenLDAP 2.5 ist jetzt auch ein fehlerfreie Replikation der Konfiguration zwischen Providern möglich. Das heißt, dass alle Konfigurationen auf allen Providern absolut identisch sind. So können Sie auch schnell einen zusätzlichen Provider einbinden, wenn die Anzahl der Provider nicht mehr ausreichend ist.

Grundlegende Einrichtung eines weiteren Providers

Die Einrichtung eines zusätzlichen Providers gestaltet sich sehr einfach:

Im ersten Schritt installieren Sie die Symas-Pakete auf dem System. Anschließend erstellen Sie die benötigten TLS-Zertifikate. Achten Sie bei den Zertifikaten darauf, dass der Name und der Speicherort der Zertifikate auf allen Providern identisch sind, denn sonst klappt die Übernahme der Konfiguration aus dem ersten Provider nicht. Stellen Sie dann in der Datei */etc/default/symas-openldap* den Start des LDAP-Servers auf die dynamische Konfiguration um. Nachdem Sie den zusätzlichen Provider so vorbereitet haben, werden jetzt noch ein paar zusätzliche Einstellungen auf den Providern benötigt.

Gleichzeitig mit der Replikation der Objektdatenbank soll die Replikation auf *delta-syncrepl* umgestellt werden. Dafür wird ein Verzeichnis mit den entsprechenden Rechten benötigt, da die Datenbank, in der die Änderungen gespeichert werden, auf der Festplatte gespeichert wird und nicht im Arbeitsspeicher. In Listing 17.77 sehen Sie die Kommandos und den von uns verwendeten Pfad für die Datenbank:

```
mkdir /var/symas/accesslog
chown openldap: /var/symas/accesslog
chmod 770 /var/symas/accesslog
```

Listing 17.77 Anlegen des Verzeichnisses für die Accesslog-Datenbank

Legen Sie das Verzeichnis auf beiden Providern an.

Sorgen Sie dafür, dass das Modul `accesslog.1a` auf allen Providern geladen wird.

Anpassung des ersten Providers

Für die *MPR* benötigen alle Provider eine eindeutige `serverID`. Die ID wird im globalen Teil der Replikation angelegt. Dazu nutzen Sie die LDIF-Datei aus Listing 17.78:

```
dn: cn=config
changetype: modify
replace: olcServerID
olcServerID: 101 ldap://ldap01.example.net
olcServerID: 102 ldap://ldap02.example.net
```

Listing 17.78 Hinzufügen der »serverID«

Auch wenn Sie noch keine `serverID` vergeben haben, können Sie hier `replace` nutzen, denn `replace` erstellt ein Attribut, wenn es noch nicht vorhanden ist.



Im nächsten Schritt werden dann alle Änderungen an der Konfiguration vorgenommen. Die LDIF-Datei aus Listing 17.79 finden Sie auch in den Download-Dateien zum Buch.:

```
dn: olcDatabase={3}mdb,cn=config
changetype: add
objectClass: olcDatabaseConfig
objectClass: olcMdbConfig
olcDatabase: {3}mdb
olcDbDirectory: /var/symas/accesslog
olcSuffix: cn=accesslog
olcAccess: {0}to dn.subtree="cn=accesslog"
  by dn.exact="uid=repl-user,ou=users,dc=example,dc=net" read
  by dn.exact="cn=admin,dc=example,dc=net" read
olcLastMod: TRUE
olcMaxDerefDepth: 15
olcReadOnly: FALSE
olcRootDN: cn=config
olcLimits: dn.exact="cn=uid=repl-user,dc=example,dc=net" time=unlimited \
size=unlimited
olcSizeLimit: unlimited
olcTimeLimit: unlimited
olcMonitoring: TRUE
olcDbCheckpoint: 0 0
olcDbIndex: entryCSN eq
olcDbIndex: entryUUID eq
olcDbIndex: objectClass eq
```

```
olcDbIndex: reqEnd eq
olcDbIndex: reqResult eq
olcDbIndex: reqStart eq
olcDbIndex: reqDN eq
olcDbMode: 0600
olcDbSearchStack: 16
olcDbMaxsize: 85899345920
```

```
dn: olcOverlay=syncprov,olcDatabase={3}mdb,cn=config
changetype: add
objectClass: olcOverlayConfig
objectClass: olcSyncProvConfig
olcOverlay: syncprov
olcSpNoPresent: TRUE
olcSpReloadHint: TRUE
```

```
dn: olcOverlay={1}accesslog,olcDatabase={2}mdb,cn=config
changetype: add
objectClass: olcOverlayConfig
objectClass: olcAccessLogConfig
olcOverlay: {1}accesslog
olcAccessLogDB: cn=accesslog
olcAccessLogOps: writes
olcAccessLogSuccess: TRUE
olcAccessLogPurge: 01+00:00 00+04:00
```

```
dn: olcDatabase={2}mdb,cn=config
changetype: modify
replace: olcSyncrepl
olcSyncrepl: rid=101
  provider=ldaps://ldap01.example.net
  bindmethod=simple
  timeout=0
  network-timeout=0
  binddn=uid=repl-user,ou=users,dc=example,dc=net
  credentials=geheim
  filter="(objectclass=*)"
  searchbase="dc=example,dc=net"
  logfilter="(&(objectClass=auditWriteObject)(reqResult=0))"
  logbase=cn=accesslog
  scope=sub
  schemachecking=off
  type=refreshAndPersist
```

```

retry="60 +"
syncdata=accesslog
keepalive=240:10:30
tls_reqcert=allow
olcSyncrepl: rid=102
provider=ldaps://ldap02.example.net
bindmethod=simple
timeout=0
network-timeout=0
binddn=uid=repl-user,ou=users,dc=example,dc=net
credentials=geheim
filter="(objectclass=*)"
searchbase="dc=example,dc=net"
logfilter="(&(objectClass=auditWriteObject)(reqResult=0))"
logbase=cn=accesslog
scope=sub
schemachecking=off
type=refreshAndPersist
retry="60 +"
syncdata=accesslog
keepalive=240:10:30
tls_reqcert=allow
-
add: olcMultiprovider
olcMultiprovider: TRUE

```

Listing 17.79 Hinzufügen der Replikation der Objektdatenbank

Was passiert hier? Im ersten Teil, `dn: olcDatabase={3}mdb,cn=config`, wird eine neue Datenbank angelegt. In dieser Datenbank werden später alle Änderungen an Objekten eingetragen. Dort sehen Sie auch, dass die Datenbank in dem vorher angelegten Verzeichnis erzeugt wird. Wichtig ist in dem Abschnitt, dass Sie Ihrem Replikationsbenutzer das Leserecht an der Datenbank geben, sonst funktioniert die Replikation nicht.

Im nächsten Teil, `dn: olcOverlay=syncprov,olcDatabase={3}mdb,cn=config`, wird der Datenbank das Overlay `syncprov` zugewiesen. Ohne dieses Overlay könnte der andere Provider nicht auf die Datenbank zugreifen. Anschließend wird im Teil `dn: olcOverlay={1}accesslog,olcDatabase={2}mdb,cn=config` das Overlay `accesslog` zur Objektdatenbank hinzugefügt.

Erst jetzt folgt der eigentliche Eintrag für die Replikation im Abschnitt `dn: olcDatabase={2}mdb,cn=config`. Beachten Sie, dass hier alle Provider eingetragen sind, auch der Provider, auf dem Sie die Replikation gerade eintragen. Da der Provider dann auf sich selbst repliziert, kann so geprüft werden, ob eine Replikation fehlerfrei durchgeführt wurde. Zum

Schluss tragen Sie noch ein, dass es sich um eine Multiprovider-Replikation handelt. Ohne diesen Eintrag funktioniert die Replikation nicht. Tragen Sie die LDIF-Datei `ldapmodify` in die Konfiguration des ersten Providers mit ein.

Wenn Sie jetzt die identische Konfiguration auf den zweiten Provider übertragen würden, würde die Objektdatenbank sofort repliziert. Aber wir wollen auch das Konfigurationsbackend `cn=config` in Zukunft replizieren, sodass Sie alle Änderungen der Konfiguration nur noch an einem Provider durchführen müssen. Die LDIF-Datei für die Replikation der Konfiguration sehen Sie in Listing 17.80:

```
dn: olcOverlay=syncprov,olcDatabase={0}config,cn=config
changetype: add
objectClass: olcOverlayConfig
objectClass: olcSyncProvConfig
olcOverlay: syncprov

dn: olcDatabase={0}config,cn=config
changetype: modify
replace: olcSyncRepl
olcSyncRepl: rid=1
  provider=ldaps://ldap01.example.net
  binddn="cn=admin,cn=config"
  bindmethod=simple
  credentials=geheim
  searchbase="cn=config"
  type=refreshAndPersist
  retry="5 5 300 20"
  timeout=1
  tls_reqcert=allow
olcSyncRepl: rid=2
  provider=ldaps://ldap02.example.net
  binddn="cn=admin,cn=config"
  bindmethod=simple
  credentials=geheim
  searchbase="cn=config"
  type=refreshAndPersist
  retry="5 5 300 20"
  timeout=1
  tls_reqcert=allow
-
add: olcMultiprovider
olcMultiprovider: TRUE
```

Listing 17.80 Die LDIF-Datei für die Replikation der Konfiguration

Da hier keine *delta-syncrepl* genutzt wird, brauchen Sie auch hier keine neue Datenbank zu erzeugen. Die LDIF-Datei besteht daher nur aus drei Teilen: Im ersten Teil, dn: `olcOverlay=syncprov,olcDatabase={0}config,cn=config`, wird das Overlay `syncprov` zur Konfigurationsdatenbank hinzugefügt. Im zweiten Teil, dn: `olcDatabase={0}config,cn=config` wird die eigentliche Replikation konfiguriert. Auch werden wieder alle Provider eingetragen.

Im letzten Teil wird auch hier wieder dem Provider mitgeteilt, dass es sich um eine Multi-provider-Replikation handelt. Auch diese LDIF-Datei spielen Sie jetzt mit `ldapmodify` in die Konfiguration ein. Jetzt sichern Sie die gesamte Konfiguration so wie Sie es in Listing 17.81 sehen, und kopieren die Sicherung auf den zweiten Provider:

```
root@ldap01:~# slapcat -n0 -F /opt/symas/etc/openldap/slapd.d > backup-config.ldif
root@ldap01:~# scp back-config-murks.ldif ldap02:
```

Listing 17.81 Sicherung der Konfiguration

Anpassung des zweiten Providers

Auf dem zweiten Provider haben Sie bis zu diesem Zeitpunkt noch keine Konfiguration eingetragen und es sind auch noch keine Objekte vorhanden. Alles das folgt jetzt in diesem Schritt. Da Sie zurzeit keine Konfiguration auf dem zweiten Provider erstellt haben, lässt sich der LDAP-Dienst auch nicht starten. Sie können aber die vom ersten Provider gesicherte Konfiguration auf dem zweiten Provider mit dem Kommando `slapadd` einspielen. In Listing 17.82 sehen Sie diesen Vorgang:

```
root@ldap02:~# slapadd -n0 -F /opt/symas/etc/openldap/slapd.d/ -l backup-config.ldif
Closing DB...
root@ldap02:~# chown -R openldap: /opt/symas/etc/openldap/slapd.d/
root@ldap02:~# systemctl restart symas-openldap-server.service
root@ldap02:~# ldapsearch -Y EXTERNAL -H ldapi:///
root@ldap02:~# ldapsearch -Y EXTERNAL -H ldapi:/// -b cn=config
```

Listing 17.82 Einspielen der Konfiguration auf dem zweiten Provider

Die beiden letzten Kommandos zeigen Ihnen den Inhalt der Objektdatenbank und der Konfiguration an. Wenn Sie jetzt Änderungen an der Konfiguration oder den Objekten vornehmen, werden alle Änderungen auch sofort auf den anderen Provider übertragen. Wenn jetzt einer der beiden Provider ausfällt, sorgt der zweiten Provider für Ausfallsicherheit.

Bis zu diesem Punkt haben Sie die Ausfallsicherheit der Provider eingerichtet. Wenn Sie jetzt noch Ihre Consumer so konfigurieren, dass für beide Provider ein *syncrepl*-Eintrag vorhanden ist, werden Ihre Consumer beim Ausfall eines Providers immer noch in der Lage sein, Änderungen zu bekommen.

17.16 Weiterleitungen für den Mailserver Postfix

Wenn Sie Ihren Mailserver so eingerichtet haben, wie in Kapitel 10, »Mailserver«, gezeigt wird und Sie Ihre User-Accounts im LDAP verwalten, können Sie Ihren Postfix mit wenigen Handgriffen auch an Ihren LDAP-Server anbinden. Am Ende kann Postfix seine Weiterleitungen und Aliasse auch direkt aus LDAP beziehen, sodass Sie keine lokalen Dateien direkt auf dem Server mehr pflegen müssen. Bei openSUSE und CentOS ist eine LDAP-Unterstützung bereits im Hauptpaket enthalten, bei Debian/Ubuntu müssen Sie das Paket `postfix-ldap` separat nachinstallieren.

Grundsätzlich können Sie alles, was Sie in Postfix als Map-Datei im Format *hash* oder *btree* anlegen können, auch direkt aus LDAP abfragen lassen. Diese Tabellen haben immer einen zweispaltigen Aufbau: Spalte eins ist die *Query*, also das Suchkriterium, Spalte zwei ist *Result/Action*, also das Suchergebnis. Bei der Abfrage von Map-Dateien liest Postfix also nie wie bei einer Konfigurationsdatei den Inhalt der Datei ein, sondern »fragt« jedes Mal erneut die jeweilige Datei oder Datenbank – die auch ein LDAP-Server sein kann.

Wenn Ihre User im LDAP zusätzliche Mailadressen im LDAP-Eintrag `mailAliases` gespeichert haben, benötigen Sie eine Weiterleitung von `mailAliases` (=Query) auf die Haupt-Mailadresse `mail` (=Result). Definieren Sie diese Abfrage zunächst in einer Datei `/etc/postfix/virtual.ldap`, so wie Sie es in Listing 17.83 sehen:

```
server_host = adminbuch.example.net
#server_port = 389
#start_tls = yes
#version = 3

bind = yes
bind_dn = uid=postfix,ou=users,dc=example,dc=net
bind_pw = geheim

search_base = dc=example,dc=net
scope = sub
query_filter = (mailAliases=%s)
result_attribute = mail

### Diese Variante schreibt Mailadressen auf die User-ID der Nutzer um.
### Der Result-Filter ergänzt die Domain für ein sauberes Mail-Routing!
# query_filter = (|(mailAliases=%s)(mail=%s))
# result_attribute = uid
# result_filter = %s@example.com
```

Listing 17.83 Die Definition der LDAP-Abfrage für »virtual_alias_maps«

Die einzelnen Parameter haben dabei folgende Bedeutung:

- ▶ `server_host = adminbuch.example.net`
Das ist der FQDN des LDAP-Servers.
- ▶ `server_port = 389`
Das ist der TCP-Port des LDAP-Servers (falls abweichend vom Default).
- ▶ `start_tls = yes`
Dieser Parameter aktiviert (optional) eine TLS-gesicherte Verbindung zum LDAP-Server.
- ▶ `version = 3`
Diese Zeile zeigt die zu verwendende Protokollversion zum LDAP-Server.
- ▶ `bind = yes`
Dieser Parameter teilt dem Postfix-Server mit, dass er sich auf jeden Fall beim LDAP authentifizieren muss und dass keine anonyme Suche erlaubt ist.
- ▶ `bind_dn = uid=postfix,ou=users,dc=example,dc=net`
Dies ist der Benutzer, mit dem sich der Postfix am LDAP authentifiziert. Wie schon bei der Replikation können Sie auch hier ein *SimpleSecurityObject* erzeugen, das nur einen Namen und ein Passwort besitzt. Damit kann sich niemand über eine Shell mit dem Konto anmelden.
- ▶ `bind_pw = geheim`
Das ist das Passwort des Benutzers für die Authentifizierung am LDAP.
- ▶ `search_base = dc=example,dc=net`
Ab diesem Punkt im DIT wird der Postfix die Suche nach den Aliassen beginnen. Hier können Sie natürlich auch einen anderen Punkt im DIT angeben: Es muss nicht, so wie in Listing 17.83, die oberste Ebene des DIT sein.
- ▶ `scope = sub`
Der Scope gibt an, wie weit der Postfix ab der *Base* suchen soll. Hier wird *sub* verwendet: Damit wird der gesamte DIT nach Alias-Einträgen durchsucht. Alternativ sind auch *base* (nur die Ebene der Base-DN) oder *one* (Base-DN und die darunterliegende Ebene) möglich.
- ▶ `query_filter = (mailAliases=%s)`
Das ist der Filter, der bei der Suche angewendet wird. Dabei setzt Postfix anstelle des %s den Suchbegriff ein, hier also die ursprüngliche Mailadresse.
- ▶ `result_attribute = mail`
Hat die Suche ein LDAP-Objekt als Treffer gefunden, liest Postfix den Inhalt dieses LDAP-Objekts als Suchergebnis.
- ▶ `result_filter = %s@example.com`
Der optionale Result-Filter ermöglicht Manipulationen und Ergänzungen am Suchergebnis. Dabei steht %s als Platzhalter für das ursprüngliche Suchergebnis aus `result_attribute`.



In der Manpage `man 5 ldap_table` finden Sie weitere Details und Parameter für LDAP-Abfragen durch Postfix. Wenn der DIT über ACLs abgesichert ist, müssen Sie über eine ACL sicherstellen, dass der `bind_dn` die Leseberechtigung an dem Attribut `mail` hat.

Nun müssen Sie Postfix noch anweisen, die LDAP-Suche aus der Datei `virtual.ldap` als Abfrage für `virtual_alias_maps` zu verwenden. Lassen Sie dabei `virtual_alias_maps` weiterhin als Erstes auf eine echte lokale Textdatei zeigen. Auch wenn diese (hoffentlich) fast immer leer ist, können Sie so im Notfall auch weiterhin schnell und unkompliziert Weiterleitungen per Hand eintragen, ohne immer gleich tatsächlich Einträge im LDAP ändern zu müssen (siehe Listing 17.84):

```
virtual_alias_maps = hash:/etc/postfix/virtual, proxy:ldap:/etc/postfix/virtual.ldap
```

Listing 17.84 Definition der »`virtual_alias_maps`«-Tabelle mit LDAP

Mit dem zusätzlichen Präfix `proxy:` veranlassen Sie, dass das `smtpd`-Modul von Postfix die Verbindung zum LDAP-Server nicht selbst aufbaut, sondern diese über das `Proxy`-Modul von Postfix führt. Auf diese Art und Weise können bestehende Verbindungen mehrfach genutzt werden, und es werden weniger zeitgleich bestehende Verbindungen zu Ihrem LDAP-Server aufgebaut.

Mit dem `postmap`-Kommando können Sie schnell und einfach prüfen, ob und welches Ergebnis Ihre definierte LDAP-Abfrage liefert. Suchen Sie sich einen Nutzer mit Aliasen/Weiterleitungen, und prüfen Sie auf dem Mailserver, ob diese auch gefunden werden (siehe Listing 17.85):

```
mail:~ # postmap -q aliasadresse@example.com ldap:/etc/postfix/virtual.ldap  
zieluser@example.com
```

Listing 17.85 Überprüfung der LDAP-Abfrage mit »`postmap`«

17.17 Benutzerauthentifizierung von Dovecot über LDAP

In Abschnitt 10.2, »POP3/IMAP-Server mit Dovecot«, haben wir den IMAP-Server Dovecot ausführlich beleuchtet und auch die grundlegenden Fragen zur Einrichtung eines IMAP-Servers erklärt. Wenn Sie Ihre Nutzer bereits in einem LDAP-Server verwalten, können Sie die Authentifizierung von Dovecot mit wenigen Handgriffen umrüsten und ebenfalls an diesen LDAP-Server binden.

Bei Debian/Ubuntu müssen Sie das Paket `dovecot-ldap` separat installieren, bei openSUSE ist die LDAP-Unterstützung im Grundpaket bereits enthalten. Ganz am Ende der Datei `/etc/dovecot/conf.d/10-auth.conf` können Sie nun die Authentifizierungsquelle an LDAP anpassen. Deaktivieren Sie alle anderen Authentifizierungsmethoden, und binden Sie stattdessen `auth-ldap.conf.ext` ein (siehe Listing 17.86):


```

#!include auth-deny.conf.ext
#!include auth-master.conf.ext
#!include auth-system.conf.ext
#!include auth-sql.conf.ext
!include auth-ldap.conf.ext
#!include auth-passwdfile.conf.ext
#!include auth-checkpassword.conf.ext
#!include auth-vpopmail.conf.ext
#!include auth-static.conf.ext

```

Listing 17.86 Die LDAP-Definition in Dovecot einbinden

Darüber wird auf die Konfigurationsdatei `/etc/dovecot/dovecot-ldap.conf` verwiesen, die nicht nur Zugangsdaten zum LDAP-Server beinhaltet, sondern auch LDAP-Suchfilter für die User- und Passwortsuche definiert. Bei der Definition der Suchfilter können Sie auf drei Variablen zurückgreifen:

- ▶ `%u`
auf die vollständige Mailadresse oder den Usernamen, also `user@example.com`
- ▶ `%n`
auf den Localpart von Mailadresse oder den Usernamen, also den Text bis zum `@`-Zeichen: `user`. Ist kein `@`-Zeichen vorhanden, ist `%u` identisch mit `%n`.
- ▶ `%d`
auf den Domainpart von Mailadresse oder den Usernamen, also den Text nach dem `@`-Zeichen: `example.com`. Ist kein `@`-Zeichen vorhanden, bleibt `%d` leer.

Durch ein zusätzliches `l` wird der Parameter *lowercase*, also »in Kleinschreibung«, verwendet. Sie sollten sich grundsätzlich `%Lu`, `%Ln` oder `%Ld` angewöhnen.

Auch wenn die Groß- und Kleinschreibung bei LDAP ignoriert wird, ist sie bei Datenbankabfragen oder bei der Definition des Homeverzeichnis des Nutzers (siehe Listing 17.87) elementar wichtig.

Haben Sie im LDAP ganz normale Userdaten in den Objektklassen `posixAccount` definiert, können Sie eine Authentifizierung mit folgenden LDAP-Einstellungen vornehmen, wenn sich Ihre User mit der User-ID anmelden sollen (siehe Listing 17.87):

```

hosts = ldap.example.com
dn = cn=dovecot,dc=example,dc=com
dnpass = <ldappasswort>
ldap_version = 3

```

```

base = ou=user,dc=example,dc=com
scope = subtree

```



```
user_attrs = homeDirectory=home, uid=uid, gid=gid
user_filter = (&(objectClass=posixAccount)(uid=%Ln))
```

```
pass_attrs = uid=user, userPassword=password
pass_filter = (&(objectClass=posixAccount)(uid=%Ln))
```

Listing 17.87 Konfiguration des LDAP-Zugriffs zum Login mittels UID

In den Parametern `dn` und `bind_dn` ist ein spezieller LDAP-Bind (Benutzer) konfiguriert, der die Berechtigung haben muss, Benutzerdaten aus dem LDAP-Server zu erfragen. Achten Sie darauf, dass dieser User im LDAP angelegt und gegebenenfalls mit den passenden ACLs versehen ist. Sie können Postfix und Dovecot hier guten Gewissens den gleichen LDAP-User verwenden lassen. Allerdings müssen Sie in diesem Beispiel dafür Sorge tragen, dass alle E-Mails, die Sie von Postfix mittels LMTP an Dovecot routen, bereits die UID als Localpart der Mailadresse (= %n) verwenden, also an `uid@example.com` adressiert sind.



Sind bei Ihnen Localpart und UID nicht identisch, müssen Sie Ihre E-Mails vorher in Postfix (wie eben in Abschnitt 17.16 gezeigt) über die `virtual_alias_map` an `uid@example.com` weiterleiten lassen.

Bequemer ist es, wenn Sie auf ein LDAP-Schema zurückgreifen können, das bereits ein `mail`-Attribut aufweist. Dann können Sie dieses Attribut direkt in den `user_filter` einbinden und die Suche nach der Mailadresse als ODER-Abfrage auf dieses Attribut erweitern. So müssen Sie die Adressen in Postfix vorher nicht umschreiben lassen:

```
user_attrs = uid=user, homeDirectory=home, uid=uid, gid=gid
user_filter = (&(objectClass=posixAccount)(|(uid=%u)(mail=%u)))
```

```
pass_attrs = uid=user, userPassword=password
pass_filter = (&(objectClass=posixAccount)(uid=%u))
```

Listing 17.88 Konfiguration des LDAP-Zugriffs zur direkten Suche der Mailadresse

Beachten Sie, dass in dieser Variante trickreich auch beim `uid`-Attribut auf `%u` zurückgegriffen wurde. Ein `%n` könnte dafür sorgen, dass ein Localpart einer Mailadresse auf eine nur zufällig identische User-ID eines anderen Anwenders zutrifft. Möchten Sie hingegen, dass sich Ihre User nicht mit ihrer User-ID, sondern stets mit ihrer vollen Mailadresse anmelden, können Sie Ihre Filter entsprechend anpassen und auf den Rückgriff auf die User-ID verzichten:

```
user_attrs = homeDirectory=home, uid=uid, gid=gid
user_filter = (&(objectClass=posixAccount)(mail=%u))
```

```
pass_attrs = userPassword=password
pass_filter = (&(objectClass=posixAccount)(mail=%u))
```

Listing 17.89 Konfiguration des LDAP-Zugriffs zum Login mit Mailadresse



Die Kommandos `doveadm user` und `doveadm auth test` helfen, wie schon in Abschnitt 10.2.4, »User-Authentifizierung«, gezeigt, beim Debugging der Userabfrage.

In Abschnitt 10.2, »POP3/IMAP-Server mit Dovecot«, haben wir Ihnen geraten, auf dem IMAP-Server für alle Anwender die stets gleiche ID 10.000 zu verwenden. Außerdem haben wir empfohlen, den Speicherpfad der E-Mails nach `/srv/vmail/<domain>/<user>` zu verlegen, um die virtuellen Mailnutzer von existierenden Shell-Nutzern klar getrennt zu halten. Sie können über einen einfachen Trick dafür sorgen, dass Dovecot diese Werte automatisch so setzt und abweichende Werte im LDAP ignoriert. Passen Sie dazu die Parameter `user_attrs` und `pass_attrs` so wie in Listing 17.90 an:

```
user_attrs = =home=/srv/vmail/%Ld/%Lu, =uid=10000, =gid=10000
pass_attrs = =userdb_home=/srv/vmail/%Ld/%Lu, =userdb_uid=10000, =userdb_gid=10000
```

Listing 17.90 Konfiguration statischer LDAP-Suchergebnisse für Dovecot

Achten Sie dabei penibel auf die mehrfach gesetzten Gleichheitszeichen. Es ist richtig, dass die Einträge für `uid`, `gid` und `home` mit einem »leeren« Gleichheitszeichen beginnen, denn davor stünde ursprünglich ja das auszulesende LDAP-Attribut.

17.18 Benutzerauthentifizierung am Proxy Squid über LDAP

17

Wenn in einem Netzwerk der Zugang zum Internet über einen Proxy mit Anmeldung realisiert werden soll, können Sie den Proxy *Squid* so einstellen, dass er LDAP nutzt, um die Benutzer zu authentifizieren. Hinweise zur Einrichtung des Proxys *Squid* finden Sie in Abschnitt 13.1, »Einführung des Stellvertreters«, in dem auch das Thema Authentifizierung intensiv behandelt wird. Es gibt zwei verschiedene Möglichkeiten für die Authentifizierung. Zum einen kann eine Authentifizierung auf Benutzerebene stattfinden und zum anderen auf Gruppenebene. Beide Möglichkeiten sollen hier angesprochen werden.

Wie in den vorherigen Abschnitten geht es nicht darum, eine komplette Konfiguration des Proxys *Squid* vorzunehmen, sondern lediglich darum, zu zeigen, wie der *Squid* auf die Benutzer oder Gruppen im LDAP zugreifen kann, um Benutzer zu authentifizieren.

17.18.1 Die Authentifizierung über LDAP aktivieren

Im ersten Schritt müssen Sie sowohl für die benutzerbezogene Authentifizierung als auch für die gruppenbezogene Authentifizierung das entsprechende Modul für die Kommunikation mit dem LDAP laden. Dazu ändern Sie bei allen Distributionen in der Datei `/etc/squid/squid.conf` die Parameter am TAG: `auth_param` auf die Werte (siehe Listing 17.91):

```
[...]
#TAG: auth_param
```

```

auth_param basic program /usr/lib/squid/ldap_auth -v 3\
    -b "dc=example,dc=net" -f "uid=%s" \
    -D "cn=admin,dc=example,dc=net" \
    -W /root/pw-file-squid localhost
auth_param basic children 5
auth_param basic realm Web-proxy
auth_param basic credentialsttl 2 minutes

```

Listing 17.91 »auth_pam«-Einträge in der »squid.conf«

Die dort eingetragenen Parameter haben folgende Bedeutung:

- ▶ `auth_param basic program`
Dieser Parameter legt das Programm fest, das für die Authentifizierung der Benutzer verantwortlich ist. Für Debian und Ubuntu ist das `/usr/lib/squid/ldap_auth`. Unter openSUSE und CentOS heißt das Programm `/usr/sbin/squid_ldap_auth`. Im Anschluss an das entsprechende Programm folgen die Parameter für das Programm. `-v 3` legt fest, dass die LDAP-Version 3 verwendet wird. `-b "dc=example,dc=net"` legt den Startpunkt der Suche fest. Mit `-f "uid=%s"` wird das Attribut festgelegt, in diesem Fall `uid`, mit dem der übergebene Benutzername aus dem Anmeldefenster verglichen wird.

Mit dem Parameter `-D "cn=admin,dc=example,dc=net"` wird der `binddn` für die Suche festgelegt. Damit Sie das Passwort nicht direkt in die `/etc/squid/squid.conf` schreiben müssen, wird hier das Passwort in die Datei `/root/pw-file-squid` geschrieben. Dafür sorgt der Parameter `-W /root/pw-file-squid`. Zum Schluss wird noch der Name des Rechners angegeben, auf dem der LDAP-Server läuft – hier der `localhost`.



Die Parameter müssen alle in einer Zeile geschrieben werden, können aber so wie hier im Beispiel durch einen Backslash umbrochen werden.

- ▶ `auth_param basic children 5`
Dieser Parameter legt fest, wie viele Authentifizierungsprozesse gleichzeitig stattfinden können. Wird der Wert zu klein gewählt, kann es zu Verzögerungen kommen, wenn sich viele Benutzer gleichzeitig anmelden. Verzögerungen kommen. Diesen Wert sollten Sie daher auf jeden Fall beobachten.
- ▶ `auth_param basic realm Web-proxy`
Der Text, der hier an den Parameter angehängt wird (in diesem Beispiel »Web-proxy«), erscheint anschließend als Text im Anmeldefenster, wenn ein Benutzer auf den Squid zugreift und sich anmelden will.
- ▶ `auth_param basic credentialsttl 2 minutes`
Dieser Parameter legt fest, wie lange eine Authentifizierung gültig ist. Wenn die Zeit kurz gesetzt wird und der Benutzer den Browser schließt, wird er schneller wieder nach dem Passwort gefragt.

17.18.2 Benutzerbezogene Authentifizierung

Nachdem nun der Squid LDAP für die Authentifizierung nutzt, soll als Erstes eine benutzerbezogene Authentifizierung eingerichtet werden. Dazu müssen Sie eine *ACL* und eine *http_access*-Regel in der *squid.conf* so wie in Listing 17.92 erstellen:

```
acl ldap-auth proxy_auth REQUIRED
```

Listing 17.92 Squid-ACL für die benutzerbezogene Authentifizierung

Um die *ACL* auch wirksam werden zu lassen, müssen Sie noch eine *http_access*-Regel erstellen. In Listing 17.93 sehen Sie die entsprechende Regel:

```
http_access allow ldap-auth
```

Listing 17.93 »http_access«-Regel für die benutzerbezogene Authentifizierung

Nach einem Neustart des Squid kann sich jeder Benutzer, der im LDAP-Baum ein Konto hat, am Proxy anmelden.

17.18.3 Gruppenbezogene Authentifizierung

Oft sollen aber unterschiedliche Benutzergruppen unterschiedliche Zugriffszeiten oder -möglichkeiten auf das Internet haben. Dazu werden Gruppen benötigt, die dann über *ACLs* Rechte erhalten. Am Beispiel der Gruppe *squidusers* soll hier gezeigt werden, wie die Authentifizierung auf Gruppenebene realisiert werden kann. Dazu müssen Sie eine *externe ACL* mithilfe des Programms *squid_ldap_group* erstellen und anschließend wieder eine *http_access*-Regel erstellen, die den Zugriff für die Gruppe regelt. In Listing 17.94 sehen Sie einen Ausschnitt aus der *squid.conf* eines Debian- oder Ubuntu-Servers. Bei openSUSE und CentOS muss der Pfad des Programms auf */usr/sbin/squid_ldap_group* angepasst werden.

```
external_acl_type ldap_filter %LOGIN /usr/lib/squid/squid_ldap_group\  
    -b "dc=example,dc=net" -v 3\  
    -f "(&(objectClass=posixGroup)(cn=%a)(memberUid=%v))\  
    -D "cn=admin,dc=example,dc=net" -W /root/pw-file-squid localhost  
acl inet_users external ldap_filter squidusers
```

Listing 17.94 *ACLs* für die gruppenbezogene Authentifizierung

Die externe *ACL* besteht eigentlich nur aus einer Zeile. Zur besseren Lesbarkeit können Sie diese Zeile aber auch wieder umbrechen. Die Parameter der *ACL* haben folgende Bedeutung:

- ▶ *ldap_filter*
das Programm, das für die Erstellung der externen *ACL* verwendet wird
- ▶ *%LOGIN*
der am Login-Fenster übergebene Benutzername

- ▶ `/usr/lib/squid/squid_ldap_group`
der Pfad zum Programm, das die ACL erstellt (Unter openSUSE und CentOS liegt das entsprechende Programm in `/usr/sbin/squid_ldap_group`.)
- ▶ `-b "dc=example,dc=net"`
der Startpunkt der Suche im LDAP-Baum
- ▶ `-v 3`
Es soll nur die LDAP-Version 3 verwendet werden.
- ▶ `-f "(&(objectClass=posixGroup)(cn=%a)(memberUid=%v))"`
Dies ist der Suchfilter für den LDAP-Baum. Geprüft wird, ob der beim `%LOGIN` übergebene Benutzer Mitglied der POSIX-Gruppe ist, die in der aufrufenden ACL angegeben wurde.
- ▶ `-D "cn=admin,dc=example,dc=net"`
legt den *binddn* fest, mit dem die Suche durchgeführt wird. Wenn hier nicht der Admin des LDAP verwendet werden soll, muss eventuell noch eine LDAP-ACL erstellt werden, die es einem eigens für den Squid erstellten Benutzer erlaubt, die Konten zu lesen.
- ▶ `-W /root/pw-file-squid`
Auch hier wird das Passwort wieder aus der Datei gelesen und nicht direkt in die *squid.conf* eingetragen.
- ▶ `localhost`
Dies ist der Name des Hosts, auf dem der LDAP-Server läuft. In diesem Fall laufen beide Dienste auf demselben Host.

In der zweiten ACL wird die vorher erstellte externe ACL zusammen mit dem Gruppennamen aufgerufen. Es wäre auch möglich, die ACL mit anderen ACLs zu verketteten, zum Beispiel mit ACLs, die die Zugriffszeiten auf das Internet regeln. Jetzt müssen Sie nur noch die `http_access`-Regel passend zu den ACLs erstellen. Die Regel sieht wie folgt aus:

```
http_access allow inet_users
```

Listing 17.95 »`http_access`«-Regel für die gruppenbezogene Authentifizierung

Nach einem Neustart des Squid können jetzt nur noch Benutzer, die Mitglied der Gruppe `squidusers` sind, auf das Internet zugreifen. Damit sind Sie in der Lage, sowohl eine benutzerbezogene als auch eine gruppenbezogene Authentifizierung der Benutzer am Proxy Squid über Ihren LDAP-Server zu regeln.

17.19 Benutzerauthentifizierung am Webserver Apache über LDAP

Als letzter Dienst soll der Apache-Webserver den LDAP-Server zur Authentifizierung von Benutzern verwenden. Dazu müssen Sie einen Apache-Webserver eingerichtet haben. Informationen zur Einrichtung des Apache-Webservers finden in Kapitel 8, »Webserver«.

Bevor Sie die Konfigurationsdatei ändern, müssen Sie noch zwei Module für den Apache laden. Unter Debian und Ubuntu heißen die Module `authnz_ldap.load` und `ldap.load`. Unter openSUSE und CentOS heißen die Module `mod_authnz_ldap.so` und `mod_ldap.so`. Nach dem Einbinden der Module müssen Sie den Webserver einmal neu starten, dann können Sie die Konfigurationsdatei anpassen.

Die Konfiguration besteht aus zwei Teilen: zum einen aus der Konfiguration der Cache-Parameter und zum anderen aus der Konfiguration des eigentlichen Zugriffs auf den LDAP-Server für die Authentifizierung. Da die Berechtigung für den Zugriff über eine Gruppe geregelt werden soll und diese auch noch möglichst automatisch mit Mitgliedern gefüllt werden soll, wird hier eine dynamische Gruppe angelegt, die aufgrund eines Attributs mit Mitgliedern gefüllt wird. Das Prinzip dazu wurde bereits bei der Erklärung der Overlays beschrieben.

Anlegen einer dynamischen Gruppe

Legen Sie eine neue *groupOfURLs* wie in Listing 17.96 an – weisen Sie jetzt einem Benutzer das Attribut `Title` mit dem Wert `Apache-user` zu, wird er automatisch Mitglied der Gruppe:

```
dn: cn=apache,ou=groups,dc=example,dc=net
objectClass: groupOfURLs
cn: apache
memberURL: ldap:///dc=example,dc=net??sub?(Title=Apache-user)
```

Listing 17.96 LDIF für die dynamische Gruppe

17

17.19.1 Konfiguration der Cache-Parameter

Die Informationen, die sich der Apache vom LDAP holt, werden in einem Cache zwischengespeichert, sodass der Apache nicht bei jeder Authentifizierung wieder auf den LDAP zugreifen muss. Die Parameter dazu finden Sie in Listing 17.97.

```
LDAPCacheEntries 1024
LDAPCacheTTL 60
LDAPOpCacheEntries 1024
LDAPOpCacheTTL 60
```

Listing 17.97 Einträge für die Cache-Parameter

Die Parameter haben folgende Bedeutung:

- ▶ `LDAPCacheEntries 1024`
Dieser Wert legt fest, wie viele erfolgreiche »search/binds« im Cache gespeichert werden. Je größer der DIT ist und je größer die Anzahl der Benutzer ist, die sich am Apache authentifizieren müssen, umso größer sollte der Wert sein, sodass eine erneute Anfrage eines Benutzers nicht immer zum LDAP-Server weitergeleitet werden muss. Das erhöht die Anmeldegeschwindigkeit und reduziert die Last auf dem LDAP-Server.

- ▶ LDAPCacheTTL 60
Dieser Wert legt die Zeit in Sekunden fest, die ein Eintrag im Cache gültig ist.
- ▶ LDAPOpCacheEntries 1024
Dieser Wert legt die Größe des zweiten Caches fest. In diesem Cache werden alle Vergleichsergebnisse abgelegt.
- ▶ LDAPOpCacheTTL 60
Auch der zweite Cache hat ein Zeitlimit für die Speicherung der Vergleichsergebnisse. Dieser Wert wird ebenfalls in Sekunden angegeben.

Ein sinnvoller Wert lässt sich hier nur durch längere Beobachtungen des Servers ermitteln. Diese Werte sollten Sie, besonders am Anfang, immer im Auge behalten.



Die Parameter für den Cache müssen unbedingt am Anfang der Konfigurationsdatei stehen. Wenn es sich bei der Konfigurationsdatei um einen virtuellen Webserver handelt, müssen die Parameter vor der Zeile `<VirtualHost 192.168.123.120:443>` stehen.

17.19.2 Konfiguration der Zugriffsparameter

Nachdem der Cache definiert wurde, können Sie nun beginnen, die eigentliche Authentifizierung durch den LDAP-Server zu konfigurieren. Dazu müssen Sie die Parameter aus Listing 17.98 in die Konfigurationsdatei des Webservers eintragen:

```
AuthType Basic
AuthBasicProvider ldap
AuthLDAPBindDN "uid=apache-user,ou=users,dc=example,dc=net"
AuthLDAPBindPassword "geheim"
AuthName "Benutzername zur Anmeldung am LAM"
AuthLDAPURL "ldap://adminbuch.example.net:389/dc=example,dc=net??sub"
require ldap-group cn=apache,ou=groups,dc=example,dc=net
AuthLDAPGroupAttributeIsDN On
AuthLDAPGroupAttribute member
```

Listing 17.98 Parameter für die eigentliche Authentifizierung

Die Parameter aus Listing 17.98 haben folgende Bedeutung:

- ▶ AuthBasicProvider ldap
Hier wird festgelegt, dass LDAP die Quelle für die »Basic-Authentifizierung« ist.
- ▶ AuthLDAPBindDN "uid=apache-user,ou=users,dc=example,dc=net"
Das ist der Benutzer, der die Suche im LDAP durchführt. Stellen Sie sicher, dass der entsprechende Benutzer das Recht besitzt, die Mitgliederliste der entsprechenden Gruppen zu lesen, und dass er Zugriff auf die Benutzerkonten zur Authentifizierung hat.
- ▶ AuthLDAPBindPassword "geheim"
Das ist das Passwort für den *AuthLDAPBindDN*.

- ▶ AuthName "Benutzername zur Anmeldung am LAM"
Dieser Text wird in der Anmeldemaske angezeigt, wenn ein Benutzer auf den Webserver zugreift.
- ▶ AuthLDAPURL "ldap://adminbuch.example.net:389/dc=example,dc=net??sub"
In dieser Zeile wird der LDAP-Server angegeben, mit dem der Webserver die Verbindung herstellen soll. In einer Zeile werden der *FQDN*, der Startpunkt der Suche (hier: *dc=example,dc=net*), das Attribut, nach dem gesucht werden soll, und die Suchtiefe festgelegt. Das Fragezeichen zwischen dem Attribut und der Suchtiefe dient hier als Trennzeichen.
- ▶ require ldap-group cn=lamadmins,ou=groups,dc=example,dc=net
Hierbei handelt es sich um die Gruppe, in der die Benutzer Mitglied sein müssen, um vom Webserver als Benutzer erkannt zu werden. Nur Mitglieder dieser Gruppe können sich am Webserver anmelden.
- ▶ AuthLDAPGroupAttributeIsDN On
Dieser Parameter sorgt dafür, dass bei den Mitgliedern auf jeden Fall ein vollständiger DN vorhanden sein muss.
- ▶ AuthLDAPGroupAttribute member
Hierbei handelt es sich um das Attribut, das in dem Gruppenobjekt gesucht wird.

Nach einem Neustart des Apache-Webservers können nur noch Mitglieder der entsprechenden Gruppe nach einer Authentifizierung auf den Webserver zugreifen.

17.20 Und was geht sonst noch alles mit LDAP?

Natürlich kann in einem Buch, das als universales »Hilfsmittel« für den fortgeschrittenen Administrator gedacht ist und das möglichst viele Dienste abdecken soll, nicht jede Möglichkeit des LDAP besprochen werden.

Neben den hier aufgeführten Techniken stehen Ihnen noch weitere Möglichkeiten offen, um den LDAP zu optimieren oder mit weiteren Diensten zu nutzen. Zu diesem Zweck ist es aber sinnvoll, ein Buch zu kaufen, das sich ausschließlich mit dem Thema LDAP befasst. Aber nach der Lektüre dieses Kapitels sollten Sie auf jeden Fall einen ersten Eindruck von der Leistungsfähigkeit und den Möglichkeiten eines LDAP-Servers gewonnen haben.

Kapitel 18

Druckserver

Es war ein großes Versprechen: das papierlose Büro. Aber in der Realität sieht es meist anders aus: Es wird immer mehr gedruckt, und der Papierbedarf steigt immer weiter. Da also niemand ohne Drucker auskommt, soll in diesem Kapitel das Drucksystem rund um CUPS etwas näher betrachtet werden. Wir gehen der Frage nach, wie Sie einen Printserver so einrichten, dass er zu Ihrer Umgebung passt.

Das *Common Unix Printing System* (CUPS) wurde ursprünglich von der Firma *Easy Software Products* entwickelt. Nachdem Apple den Hauptentwickler des Projekts angestellt hatte, fielen die Rechte von CUPS an *Apple*. CUPS steht aber, neben einigen proprietären Lizenzen, weiterhin unter der GPL.

CUPS ist als Client-Server-Applikation konzipiert. Der Client nimmt die Druckaufträge entgegen und sendet diese an den Server, der dann die Druckaufträge an den entsprechenden Drucker weiterleitet. Alle Druckaufträge, die der Client zum Server schickt, landen in einem Scheduler, der die Druckaufträge in das PostScript-Format umwandelt. Die so entstandenen PostScript-Formate werden an ein Backend gesendet. Dieses Backend kann den Druckauftrag entweder direkt auf einem PostScript-Drucker ausdrucken oder vor dem Drucken den Druckauftrag erst noch in die entsprechende Druckersprache übersetzen oder als *Raw-Daten* über das Netzwerk an einen anderen CUPS-Server übertragen. Diese *Raw-Daten* werden erst auf dem entfernten CUPS-Server für den entsprechenden Drucker umgewandelt. Das hat den Vorteil, dass der Client die Druckaufträge immer im PostScript-Format senden kann und sich nicht um die einzelnen Druckertypen kümmern muss.

Der modulare Aufbau von CUPS sorgt dafür, dass es die verschiedensten Daten (Texte, Bilder oder PDFs) verarbeiten kann. Deswegen könnten Sie CUPS eigentlich sehr gut in einer heterogenen Umgebung verwenden. Wenn Sie es zusammen mit Samba in Ihrem Netzwerk einsetzen, konnten Sie CUPS auch als Printserver für Windows-Clients nutzen. Dieser Printserver, der aus einer Kombination von Samba und CUPS besteht, ist in der Lage, die Druckertreiber für die Windows-Clients bereitzustellen. In der Vergangenheit war die Konfiguration, recht einfach einzurichten. Inzwischen ist die Einrichtung immer komplexer geworden: Grund sind die immer wieder auftretende Sicherheitsprobleme im Windows-Umfeld beim Thema Drucken. Es gibt es immer neue Sicherheitsfunktionen und Konfigurationsänderungen von Microsoft. Das sorgt dafür, dass der Aufwand, der getrieben werden muss, um einen Druckserver auf einem Linux-System einzurichten, immer weiter wächst.

Es müssen nicht nur die passenden Druckertreiber zur Verfügung stehen, sondern die Druckertreiber müssen zudem auch noch von Microsoft signiert sein. Aufgrund der *Print-Nightmare*-Problematik sind verschiedene Gruppenrichtlinien notwendig, um Treiber verteilen zu können. Aus diesen Gründen wollen wir an dieser Stelle keinen Printserver mit Samba mehr einrichten. Uns geht es noch nicht einmal so sehr darum, dass viele Schritte für die komplexe Konfiguration notwendig sind. Das größere Problem ist, dass grundlegende Änderungen in den letzten Jahren so häufig vorkamen, dass ein stabiler Produktivbetrieb kaum realistisch ist. Wir könnten auch nicht gewährleisten, dass eine Beschreibung der Konfiguration mit dem nächsten Windows-Update noch funktionieren würde.

18.1 CUPS administrieren

Die Administration und Konfiguration von CUPS kann sehr einfach über den integrierten Webserver realisiert werden. Dadurch lässt sich der CUPS-Server auch vom eigenen Arbeitsplatz aus einfach über einen Browser verwalten. Ein weiterer Vorteil von CUPS ist der, dass es das *Internet Printing Protocol* (IPP) für die Datenübertragung verwendet. Das IPP gibt Ihnen die Möglichkeit, über Policies den Zugriff auf die Administration und die Drucker zu regeln. Seit der CUPS-Version 1.2 können Sie die Policies genau an Ihre Umgebung anpassen. In älteren Versionen waren viele der Policies fest im CUPS verankert und konnten nur schwer verändert werden. Da die verschiedenen Distributionen unterschiedliche Versionen von CUPS einsetzen, sehen Sie in Tabelle 18.1 eine Übersicht über die verwendeten Versionen.

Distribution	CUPS-Version
Debian 11	2.23.3
Ubuntu 22.04	2.4.1
Suse	2.4.2

Tabelle 18.1 Distributionen und verwendete CUPS-Versionen

Auch im Hintergrund hat sich sehr viel verändert. Alle Neuerungen der Versionen können Sie unter www.cups.org/documentation.php nachlesen und können auch die Release-Notes zu den einzelnen Versionen. Dort können Sie auch für Ihre Version nachlesen, welche Funktionen unterstützt werden und welche nicht.

Nachdem Sie die CUPS-Pakete auf Ihrem System installiert haben, können Sie CUPS über die Policies verwalten. Die Verwaltung können Sie dabei mittels eines Texteditors oder über einen Webzugriff vornehmen. Auch innerhalb des Webzugriffs finden Sie einen Editor, der direkt auf die Konfigurationsdatei zugreifen kann. Neben den Policies müssen Sie auch immer den Netzwerkzugriff auf CUPS einrichten.

Um über das Web-Frontend auf den CUPS-Server zugreifen zu können, nehmen Sie zunächst zwei Einstellungen in der Datei *cupsd.conf* vor. Diese Einstellungen sind bei allen Distributionen identisch. CUPS ist am Anfang so konfiguriert, dass der Dienst nur über die IP-Adresse 127.0.0.1:631 erreichbar ist. Damit Sie das CUPS-Web-Frontend von Ihrem Arbeitsplatz aus aufrufen können, öffnen Sie den CUPS-Port 631 auch noch für die eigentliche IP-Adresse des Servers. Dazu editieren Sie die Datei *cupsd.conf* so wie in Listing 18.1:

```
Listen localhost:631
Listen 192.168.56.50:631
```

Listing 18.1 Aktivieren der IP-Adresse in der »cupsd.conf«

Durch die zweite Zeile, in der Sie die IP-Adresse des CUPS-Servers eintragen, wird CUPS nach einem Neustart auch über das Netzwerk erreichbar sein. Wenn Sie über den Browser versuchen, das Web-Frontend von CUPS zu erreichen, ist der Zugriff immer noch verboten. Um auf das Web-Frontend zugreifen zu können, passen Sie die folgenden Policies in der *cupsd.conf* so wie in Listing 18.2 mit einem Editor an:

```
# Restrict access to the server...
<Location />
  Order allow,deny
  allow from 192.168.56.*
</Location>

# Restrict access to the admin pages...
<Location /admin>
  Order allow,deny
  allow from 192.168.56.1
</Location>

# Restrict access to configuration files...
<Location /admin/conf>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
  allow from 192.168.56.1
</Location>

# Restrict access to log files...
<Location /admin/log>
  AuthType Default
  Require user @SYSTEM
  Order allow,deny
```

```
allow from 192.168.56.1
</Location>
```

Listing 18.2 Anpassung der Policy für den Webzugriff

Durch die erste Policy erhalten Sie überhaupt erst den Zugriff auf das Web-Frontend. Sie können mit dieser Policy zwar auf die erste Seite des Web-Frontends zugreifen; versuchen Sie aber, den CUPS-Server zu konfigurieren, erscheint immer noch eine Zugriffsfehlermeldung.

Erst wenn Sie auch die zweite Policy angepasst haben, können Sie auf den Administrationsbereich des CUPS-Servers zugreifen. Sie können hier auch wieder mit dem Stern als Jokerzeichen arbeiten, um ganzen Subnetzen den Zugriff zu erteilen. Genauso ist es aber auch möglich, mehrere `allow from`-Zeilen mit einzelnen IP-Adressen zu nutzen, um den Zugriff nur von bestimmten IP-Adressen zuzulassen.

Aber erst die dritte Policy gewährt Ihnen das Recht, den CUPS-Server zu verwalten. Wenn Sie anschließend einen neuen Drucker über das Web-Frontend eintragen möchten oder bestehende Konfigurationen verändern wollen, müssen Sie sich erst authentifizieren. Als Authentifizierung verwenden Sie hier am Anfang immer den lokalen `root`.

Mit der vierten Policy ermöglichen Sie den Zugriff auf die Logs des CUPS-Servers. Auch hier ist eine Authentifizierung notwendig.



An dieser Stelle wurde nur die IP-Adresse der eigenen Arbeitsstation eingetragen. Später werden noch andere Möglichkeiten angesprochen.

Jetzt können Sie über einen Browser auf den CUPS-Server zugreifen. Geben Sie dafür in Ihrem Browser die URL `http://<ip-adresse>:631` ein. Sie kommen dann auf die Startseite des CUPS-Servers (siehe Abbildung 18.1).

Abbildung 18.1 Startseite von CUPS

Klicken Sie anschließend auf den Karteireiter VERWALTUNG, und Sie kommen auf die Verwaltungsseite des CUPS (siehe Abbildung 18.2).

Abbildung 18.2 Verwaltungsseite von CUPS

Bis zu diesem Zeitpunkt haben Sie immer noch eine http-Verbindung. Wenn Sie jetzt auf DRUCKER HINZUFÜGEN klicken, werden Sie auf eine https-Seite weitergeleitet. Das Zertifikat für https bringt der CUPS-Server selbst mit. Da es sich hierbei um ein selbst signiertes Zertifikat handelt, bestätigen Sie die Ausnahme für das Zertifikat in Ihrem Browser. Nachdem Sie das Zertifikat bestätigt haben, werden Sie über https auf die Verwaltungsseite weitergeleitet.

Wenn Sie eigene Zertifikate nutzen wollen, finden Sie die Zertifikate für CUPS im Verzeichnis `/etc/cups/ssl`.

Um jetzt einen neuen Drucker anzulegen, klicken Sie auf DRUCKER HINZUFÜGEN. Als Erstes sehen Sie dann ein Pop-up-Fenster wie in Abbildung 18.3, in dem Sie sich als `root` authentifizieren. Erst dann können Sie in den nachfolgenden Fenstern Ihren Drucker eintragen.

Abbildung 18.3 Authentifizierung auf der Verwaltungsseite von CUPS





CUPS sucht sofort nach Netzwerkdruckern und zeigt die gefundenen Drucker auch sofort in der Auswahl an (siehe Abbildung 18.4).

CUPS.org Startseite Verwaltung Klassen Hilfe Aufträge Drucker

Drucker hinzufügen

Drucker hinzufügen (Schritt 1/5)

Lokale Drucker: CUPS-BRF (Virtual Braille BRF Printer)

Gefundene Netzwerkdrucker: Brother HL-L3230CDW series (Brother HL-L3230CDW series)

Andere Netzwerkdrucker: Backend Error Handler
 Internet Printing Protocol (http)
 Internet Printing Protocol (https)
 AppSocket/HP JetDirect
 Internet Printing Protocol (ipp)
 Internet Printing Protocol (ipps)
 LPD/LPR-Host oder -Drucker

Abbildung 18.4 Anzeige der gefundenen Drucker

Wählen Sie einen der aufgeführten Drucker aus, wird Ihnen im nächsten Schritt auch gleich die passende Netzwerkverbindung eingetragen. Diesen Wert können Sie direkt übernehmen. Wird Ihr Drucker nicht angezeigt, stellen Sie fest, wie Sie den Drucker erreichen können und tragen die entsprechende Verbindung ein. Wenn der Drucker nicht automatisch gefunden wird, heißt das nicht, dass Sie diesen Drucker nicht nutzen können. Oft ist das der Fall, wenn der Drucker keinen eigenen Printserver hat, sondern an einen externen Printserver angeschlossen ist.

Aktivieren Sie an dieser Stelle auch die Option DRUCKER IM NETZWERK FREIGEBEN, denn nur so kann der Drucker später über CUPS genutzt werden. Alle Einstellungen sehen Sie in Abbildung 18.5.

CUPS.org Startseite Verwaltung Klassen Hilfe Aufträge Drucker

Drucker hinzufügen

Drucker hinzufügen (Schritt 3/5)

Name:
(Darf alle druckbaren Zeichen außer "/", "#", und Leerzeichen enthalten)

Beschreibung:
(Menschenlesbare Beschreibung wie etwa "HP LaserJet mit Duplexer")

Ort:
(Menschenlesbarer Ort wie etwa "Labor 1")

Verbindung:

Freigabe: Drucker im Netzwerk freigeben

Abbildung 18.5 Verbindungseinrichtung des Druckers

In den nächsten Schritten wählen Sie den Druckerhersteller und das Modell des Druckers aus. Sie können auch eine eigene *PPD*-Datei einbinden. Viele Druckerhersteller bieten mittlerweile die entsprechenden Dateien für Ihre Drucker an, entweder im Internet oder auf einer mitgelieferten Treiber-CD, sofern noch vorhanden.

Bei dem hier verwendeten Brother-Drucker erhalten Sie den Treiber für Linux entweder als *.deb*- oder *.rpm*-Paket. Sie können die Pakete entpacken und dann dort die *PPD*-Datei finden.

Nachdem Sie alle folgenden Seiten mit den Einstellungen zu Ihrem Drucker ausgefüllt haben, erscheint nach ein paar Sekunden die Übersichtsseite zu Ihrem Drucker, so wie Sie es in Abbildung 18.6 sehen.



Abbildung 18.6 Zusammenfassung des neuen Druckers

18.2 Policies

Alle Berechtigungen werden bei CUPS immer über Policies geregelt. Die Policies sind Bestandteil des IPP. Mit ihnen können Sie Zugriffe auf die Administration und die Drucker steuern. Die gesamte Verwaltung der Policies wird immer in der Datei */etc/cups/cupsd.conf* durchgeführt. Diese Datei können Sie entweder direkt mit einem Editor bearbeiten oder über das Web-Frontend von CUPS. Mithilfe der Policies können Sie den Zugriff auf CUPS nur für bestimmte Gruppen zulassen, oder der Zugriff wird nur nach einer Anmeldung mit Benutzernamen und Passwort zugelassen. Auch um die Zugriffe nur von bestimmten Systemen aus möglich zu machen, gibt es verschiedene Konfigurationsmöglichkeiten. Bei den Policies wird zwischen *Operation-Policies* und *Location-Policies* unterschieden.

Die *Operation-Policies* dienen dazu, Rechte an Druckern und Aufgaben zu vergeben. Die *Operation-Policies* werden immer erst nach den *Location-Policies* abgearbeitet. Somit können über die *Operation-Policies* keine zusätzlichen Rechte vergeben werden, sondern nur die Rechte weiter eingeschränkt werden. Über die *Location-Policies* wird der eigentliche Zugriff auf eine Ressource gesteuert. Hier können Sie festlegen, wer überhaupt auf einen Drucker oder die Administration des CUPS-Servers zugreifen darf.

18.2.1 Location-Policies

In den Grundeinstellungen für den CUPS-Server wurden ja bereits vier Policies, die sogenannten *Location-Policies*, eingerichtet, um über das Netzwerk auf ihn zugreifen zu können. Jetzt soll anhand von einem Beispiel eine weitere *Location-Policy* konfiguriert werden. Bevor Sie allerdings eine Policy erstellen können, legen Sie erst über das Web-Frontend unter dem Karteireiter VERWALTUNG einen Drucker an.

Mehr zum Thema Anlegen von neuen Druckern und Druckerklassen finden Sie in Abschnitt 18.3, »Drucker und Klassen einrichten und verwalten«.

Nachdem Sie über das Web-Frontend einen Drucker eingerichtet haben, soll über eine *Location-Policy* der Zugriff auf diesen Drucker eingeschränkt werden. Dafür wird am Ende der Liste aller *Location-Policies* eine neue Policy so wie in Listing 18.3 angelegt:

```
<Location /printers/HL-L3230CDW>
  Order Deny,Allow
  Allow from @LOCAL
  Allow from 192.168.*
  Allow from 172.16.1.45
  Deny from 192.168.57.*
  Require group buch-lg
  Require user stefan
</Location>
```

Listing 18.3 Zugriffsbeschränkung für einen Drucker

Hier wurde der Zugriff auf einen Drucker eingeschränkt. Die einzelnen Parameter haben dabei folgende Bedeutung:

- ▶ `<Location /printers/HL-L3230CDW>`
An diesem Punkt beginnt die Einschränkung für den entsprechenden Drucker. Alle Drucker werden immer unter dem Punkt `/printers/` verwaltet. Im Anschluss daran folgt der Druckername, der bei der Einrichtung des Druckers verwendet wurde.
- ▶ `Order Deny,Allow`
Diese Zeile legt die Abarbeitung der Deny- und Allow-Regeln fest. Hier gibt es zwei Möglichkeiten für die Reihenfolge:
 - `Deny,Allow`
Bei dieser Einstellung ist Deny die Standardeinstellung. Als Erstes werden die Allow-Zeilen abgearbeitet und anschließend die deny-Zeilen.
 - `Allow,Deny`
Hier ist es genau umgekehrt: Die Standardeinstellung ist Allow. Als Erstes werden jetzt die Deny-Zeilen abgearbeitet und dann die Allow-Zeilen.

Aufgrund dieser Abarbeitungsregel ist es wichtig, dass Sie die anschließenden Zugriffsregeln in der richtigen Reihenfolge schreiben. Sonst kann es Ihnen passieren, dass Sie einer bestimmten IP-Adresse den Zugriff verweigern wollen, aber der Zugriff später über eine Allow-Regel wieder gewährt wird.

- ▶ Allow from @LOCAL
Die Gruppe @LOCAL ist ein fester Begriff, der alle IP-Adressen aus demselben Subnetz anspricht, in dem sich auch der CUPS-Server befindet. Alle IP-Adressen aus dem eigenen Subnetz bekommen über diese Regel Zugriff auf den Drucker.
- ▶ Allow from 192.168.*
Zusätzlich zu dem eigenen Subnetz wird der Zugriff für alle Hosts erlaubt, deren IP-Adresse mit 192.168 beginnt. Hier reicht es aus, wenn Sie einen Stern für die letzten beiden Oktette der IP-Adresse angeben.
- ▶ Allow from 172.16.1.45
Hier wird genau ein Rechner mit seiner IP-Adresse zugelassen.
- ▶ Deny from 192.168.57.*
Obwohl ja vorher dem gesamten Subnetz 192.168.* der Zugriff gewährt wurde, wird jetzt der Zugriff für ein ganz bestimmtes Subnetz nachträglich wieder gesperrt.
- ▶ Require group buch-lg
Der Zugriff wird durch diese Zeile nur für Benutzer der Gruppe buch-lg erlaubt.
- ▶ Require user stefan
Zusätzlich zu der vorher festgelegten Gruppe kann der Benutzer stefan auch auf den Drucker zugreifen.

Hier im Beispiel gibt es jetzt zum einen die Beschränkung auf IP-Adressen und zum anderen Beschränkungen auf Benutzer, die diesen Drucker nutzen dürfen. Der Benutzer stefan oder die Mitglieder der Gruppe buch-lg haben aber nur Zugriff auf den CUPS-Server, wenn der Host, an dem sie angemeldet sind, eine der erlaubten IP-Adressen hat.

18.2.2 Operation Policies

Bei den Operation-Policies geht es darum, bestimmte Einstellungen und Beschränkungen auf einen oder alle Drucker einzurichten. Bei den Operation-Policies spricht man auch oft von den *IPP-Limits*. Mithilfe dieser Limits können Sie verschiedene Funktionen konfigurieren. Nach der Installation von CUPS gibt es in der *cupsd.conf* immer schon eine `<Policy default>`. In dieser Policy werden die meisten Operationen bereits abgedeckt. Sie können eine eigene Policy erstellen oder aber die bestehende Policy abändern. Wenn Sie die `<Policy default>` für die meisten Drucker übernehmen wollen und nur für einige bestimmte Drucker eine eigene Policy erstellen möchten, empfiehlt es sich, die `<Policy default>` zu kopieren, anschließend den Namen zu ändern und dann diese Policy anzupassen. Bei den

Policies können Sie wieder dieselben Einschränkungen verwenden wie schon vorher bei den *Location-Policies*. Am Beispiel des Limits für die Job-Kontrolle soll hier die Erweiterung der Policies erklärt werden. Betrachten Sie dazu das Listing 18.4:

```
<Limit Cancel-Job CUPS-Authenticate-Job>  
  Require user @OWNER @SYSTEM stefan  
  Require group printeradmins  
  Allow from 192.168.123.*  
  Order deny,allow  
</Limit>
```

Listing 18.4 Erweiterung der Policy für die Job-Kontrolle

Hier wird das Abbrechen von Druckaufträgen nur noch aus dem Subnetz 192.168.123.0/24 erlaubt. Nur die Benutzer der Gruppe *printeradmins*, der Benutzer *stefan* und die beiden CUPS-internen Gruppen *@OWNER* und *@SYSTEM* dürfen Jobs aus der Warteschlange entfernen. Hinter der internen Gruppe *@OWNER* verbirgt sich immer der Absender eines Druckauftrags. Die Mitglieder der internen Gruppe *@SYSTEM* werden durch den Parameter `SystemGroup sys root` festgelegt.

18.2.3 Weitere Konfigurationsmöglichkeiten

Außer den Policies können Sie noch weitere Limits für den CUPS-Server setzen. Diese Limits können Sie im oberen Bereich der *cupsd.conf* eintragen, und zwar vor den Location-Policies. Diese Limits sind für den gesamten CUPS-Server relevant. Alle Limits, die Sie dort eintragen können, sind in der Online-Hilfe des CUPS-Servers verfügbar. An dieser Stelle sollen nur ein paar wichtige Limits angesprochen werden:

- ▶ **MaxClients**
Hier können Sie festlegen, wie viele Clients maximal gleichzeitig auf den CUPS-Server zugreifen dürfen. Die Standardeinstellung ist hier `MaxClients 100`.
- ▶ **MaxClientsPerHost**
`MaxClientsPerHost` legt fest, wie viele Verbindungen jeder Client gleichzeitig zum CUPS-Server aufgebaut haben darf. Wenn Sie diesen Wert nicht setzen, wird immer der Wert gesetzt, der auch bei dem Parameter `MaxClients` gesetzt ist.
- ▶ **MaxCopies**
Dieser Wert legt fest, wie viele Kopien eines Druckjobs erstellt werden dürfen. Der Standardwert ist `MaxCopies 9999`.
- ▶ **MaxJobsPerUser**
Mit diesem Wert können Sie festlegen, wie viele Jobs ein einzelner Benutzer gleichzeitig aktiv in der Warteschlange haben darf. Es gibt keinen Standardwert: Jeder Benutzer kann also beliebig viele Druckjobs an den CUPS-Server schicken.

- ▶ **MaxJobs**
Mit diesem Wert legen Sie fest, wie viele Jobs maximal gespeichert werden. Wird die maximale Anzahl erreicht, wird der älteste nicht mehr aktive Job gelöscht. Wenn noch alle Jobs in der Warteschlange aktiv sind, werden keine weiteren Jobs angenommen. Der Standardwert ist `MaxJobs 500`.
- ▶ **PreserveJobHistory**
Die Standardeinstellung ist `PreserveJobHistory On`. Damit werden alle Jobs in einer History verwaltet. Die Liste bleibt so lange erhalten, bis der Administrator den Parameter auf `PreserveJobHistory Off` stellt.
- ▶ **PreserveJobFiles**
Dieser Parameter ist standardmäßig auf `PreserveJobFiles Off` eingestellt. Wenn Sie diesen Parameter auf `PreserveJobFiles On` setzen, werden alle Druckaufträge in der Warteschlange gespeichert. Die Benutzer oder der Systemadministrator haben so die Möglichkeit, einen schon abgeschlossenen Druck erneut zu starten. Sie sollten zusätzlich auch den Parameter `MaxJobs` beachten.

Wenn Sie diesen Parameter auf `PreserveJobFiles On` setzen, sollten Sie auf jeden Fall das Verzeichnis `/var/spool/` auf eine eigene Partition auslagern, um ein Überlaufen der `root`- oder `var`-Partition zu verhindern.



- ▶ **TempDir**
Wenn Sie kein anderes Verzeichnis als `TempDir /var/spool/cups/tmp` für die Speicherung der Druckaufträge verwenden wollen, können Sie mit diesem Parameter ein anderes Verzeichnis festlegen. Wenn Sie das Verzeichnis ändern, achten Sie darauf, dass das *Sticky-Bit* in den Berechtigungen gesetzt ist. Wenn das Sticky-Bit nicht gesetzt ist, könnten alle Benutzer alle Druckaufträge löschen. Mit dem gesetzten Sticky-Bit sind die Benutzer nur noch in der Lage, die eigenen Druckaufträge zu löschen. In Listing 18.5 sehen Sie, wie die Berechtigungen auf das Verzeichnis bei allen Distributionen gesetzt sind:

```
printserver:~# ls -l /var/spool/cups/
insgesamt 4
drwxrwx--T 2 root lp 4096  2. Mai 15:36 tmp
```

Listing 18.5 Berechtigungen für »`/var/spool/cups/`«

Hier sehen Sie auch, dass nur Mitglieder der Gruppe `lp` überhaupt Rechte an dem Verzeichnis haben und somit auch nur die Gruppenmitglieder der Gruppe `lp` drucken können. Um nun die Druckumgebung mit den Parametern anzupassen, können Sie die Parameter alle hintereinander, so wie in Listing 18.6, in die `cupsd.conf` eintragen:

```
# Limits
MaxClients 50
MaxClientsPerHost 10
MaxCopies 5
```

```
MaxJobsPerUser 3
MaxJobs 100
PreserveJobHistory On
PreserveJobFiles On
```

Listing 18.6 Liste der Konfigurationsparameter

Nachdem Sie die *cupsd.conf* angepasst haben, starten Sie den CUPS-Server neu, um die Änderungen wirksam werden zu lassen. Wenn Sie die Änderungen über den eingebauten Editor in dem Web-Frontend durchführen, wird nach dem Speichern der CUPS-Server automatisch neu gestartet.

18.2.4 Browsing

Alle 30 Sekunden sendet der CUPS-Server seine Informationen ins eigene Subnetz, um alle Drucker bekannt zu geben. Diese Zeit können Sie über den Parameter `BrowseInterval` verändern. Die Zeit wird dort in Sekunden angegeben.

Wenn Sie nun einen zentralen CUPS-Server in Ihrem Netzwerk einrichten wollen und die Drucker auch über die Netzwerkgrenzen hinaus bekannt sein sollen, können Sie dieses Vorhaben über das Browsing realisieren. Dafür benötigen Sie lediglich in jedem Subnetz einen CUPS-Server, den Sie als *BrowseRelay* konfiguriert haben.

Das *BrowseRelay* gibt alle Informationen des CUPS-Servers, auf dem die Drucker konfiguriert sind, an das Subnetz weiter, in dem es sich befindet. In Listing 18.7 sehen Sie ein Beispiel für eine Browserkonfiguration:

```
BrowsePoll 192.168.123.120:631
BrowseRelay 127.0.0.1 10.10.255.255/16
BrowseInterval 60
```

Listing 18.7 »BrowseRelay«-Konfiguration

Der CUPS-Server, der die Drucker bereitstellt, hat die IP-Adresse 192.168.123.120. Das *BrowseRelay* befindet sich im Netz 10.10.0.0/16. Nachdem Sie die Einträge in der *cupsd.conf* erstellt haben, dauert es die über den Parameter `BrowseInterval` eingestellte Zeit, bis die Drucker des CUPS-Servers 192.168.123.120 im gesamten Subnetz 10.10.0.0 bekannt gegeben werden.

18.3 Drucker und Klassen einrichten und verwalten

Mit CUPS können Sie sowohl einzelne Drucker als auch Druckerklassen verwalten. Eine Druckerklasse ist eine Gruppe von identischen Druckern, die in einer druckintensiven Umgebung nach außen hin als ein Drucker dargestellt werden. Sendet ein Benutzer einen Druckauftrag an eine Klasse, wird der Druckauftrag von einem gerade freien Drucker der Klasse

abgearbeitet. Die Einrichtung von Druckern und Klassen erfolgt am einfachsten über das Web-Frontend. Je nachdem, welche Version von CUPS Sie installiert haben, unterscheidet sich die Oberfläche. Die einzelnen Schritte sind aber immer dieselben.

18.3.1 Drucker einrichten

Nachdem Sie einen neuen Drucker angelegt haben, finden Sie alle Informationen zu ihm in der Datei `/etc/cups/printers.conf`. In dieser zentralen Datei werden alle von diesem CUPS-Server verwalteten Drucker eingetragen. Listing 18.8 zeigt ein Beispiel für einen Drucker:

```
root@printserver:~# cat /etc/cups/printers.conf
# Printer configuration file for CUPS v2.3.3op2
# Written by cupsd
# DO NOT EDIT THIS FILE WHEN CUPSD IS RUNNING
NextPrinterId 2
<Printer HL-L3230CDW>
PrinterId 1
UUID urn:uuid:1638e36d-10db-35cc-5a4f-a8d228e547dc
Info Brother HL-L3230CDW series
Location
MakeModel Brother HL-L3230CDW CUPS
DeviceURI lpd://BRWEC5C687B1A7B/BINARY_P1
State Idle
StateTime 1672232444
ConfigTime 1672232462
Type 8392796
Accepting Yes
Shared Yes
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
OpPolicy default
ErrorPolicy retry-job
</Printer>
```

Listing 18.8 Auszug aus der »printers.conf«

In diesen Einträgen finden Sie neben den reinen Informationen wie Name und Standort auch den für den Drucker ausgewählten Treiber und den Status des Druckers.

Führen Sie keine Änderungen an den hier eingetragenen Druckern über einen Editor durch. Auch das Entfernen von Druckern sollten Sie grundsätzlich über das Web-Frontend durchführen. Der Grund ist der: CUPS verwaltet die Drucker für sich intern über Dateien im Ver-



zeichnis `/var/cache/cups`. Löschen Sie jetzt einen Drucker aus der Datei, stimmt für CUPS die Konfiguration nicht mehr, was dazu führen kann, dass Sie keine neuen Drucker mehr anlegen oder bestehende Drucker nicht mehr löschen können.

18.3.2 Klassen einrichten

Für eine druckintensive Umgebung kann es sinnvoll sein, mehrere Drucker zu einer Klasse zusammenzufassen und diese Klasse von Druckern anschließend den Mitarbeitern als einen Drucker zur Verfügung zu stellen. Der erste freie Drucker einer Klasse würde den nächsten Druckauftrag in der zugeordneten Warteschlange abarbeiten. Bei den Druckern sollte es sich aber um identische Drucker handeln, und die Drucker sollten alle am selben Ort stehen.

Um eine Druckerklasse zu erstellen, legen Sie erst alle Drucker an, die Mitglied der Klasse werden sollen. Im Anschluss legen Sie unter dem Karteireiter VERWALTUNG eine neue Druckerklasse an. Dort sehen Sie auch alle Drucker, die Sie bereits angelegt haben. Markieren Sie alle Drucker, die Mitglied der Klasse werden sollen, und geben Sie der Klasse einen Namen. Jetzt können Sie die Klasse erzeugen. Die neue Klasse finden Sie in der Datei `/etc/cups/classes.conf`. In Listing 18.9 sehen Sie einen Auszug aus der Datei:

```
<Class Buchhaltung>
Info
Location
State Idle
StateTime 1281524426
Accepting Yes
Shared Yes
JobSheets none none
Printer hl1430
Printer hl1430b
QuotaPeriod 0
PageLimit 0
KLimit 0
OpPolicy default
ErrorPolicy retry-current-job
</Class>
```

Listing 18.9 Auszug aus der »`/etc/cups/classes.conf`«

Wenn sich nun ein Benutzer mit der Klasse verbinden will, wird der Name der Druckerklasse, den Sie vergeben haben, als Name der Warteschlange angegeben. Wenn Samba auf den CUPS-Server Zugriff hat, wird die Warteschlange auch in Samba als Drucker angezeigt. Somit können sich auch Windows-Clients mit der Warteschlange verbinden und die Klasse nutzen.

18.4 Druckerquotas

Manchmal wollen Sie vielleicht den Zugriff auf bestimmte Drucker oder Klassen einschränken. Dazu gibt es die Druckerquotas, die Sie für einen Drucker oder eine Klasse einrichten können. Sie haben zwei Möglichkeiten der Einschränkung: Sie können die maximale Größe der Druckaufträge in einem Zeitraum beschränken oder aber die Anzahl der Seiten innerhalb eines Zeitraums. Die Einschränkung legen Sie über das Kommando `lpadmin` fest, so wie Sie es in Listing 18.10 sehen:

```
root@printserver:~# lpadmin -p HL-L3230CDW -o job-quota-period=604800 \
-o job-k-limit=2048

root@adminbuch:~# lpadmin -p Farblaser -o job-quota-period=86400 \
-o job-page-limit=20
```

Listing 18.10 Quotas für Drucker festlegen

Mit dem ersten Kommando wird für den Drucker HL-L3230CDW ein Joblimit von 2 MB pro Woche festgelegt. Im zweiten Beispiel gibt es einen Farblaserdrucker, auf dem die Benutzer nur maximal 20 Seiten pro Tag drucken dürfen.

Alle Beschränkungen sind immer für alle Benutzer gültig. Unterschiedliche Quotas können Sie nur über eine zweite Druckerdefinition realisieren, die Sie anschließend über die Zugriffsbeschränkung nur bestimmten Mitarbeitern zuweisen, für die Sie andere Quotas setzen. In Listing 18.11 sehen Sie noch ein Beispiel für den Drucker, wie die Quotas in der Datei `printers.conf` eingetragen werden:

```
root@printserver:~# cat /etc/cups/printers.conf

# Printer configuration file for CUPS v2.3.3op2
# Written by cupsd
# DO NOT EDIT THIS FILE WHEN CUPSD IS RUNNING
NextPrinterId 2
<Printer HL-L3230CDW>
PrinterId 1
UUID urn:uuid:1638e36d-10db-35cc-5a4f-a8d228e547dc
Info Brother HL-L3230CDW series
Location
MakeModel Brother HL-L3230CDW CUPS
DeviceURI lpd://BRWEC5C687B1A7B/BINARY_P1
State Idle
StateTime 1672232444
ConfigTime 1672237251
Type 8392796
Accepting Yes
```

```
Shared Yes
JobSheets none none
QuotaPeriod 604800
PageLimit 0
KLimit 2048
OpPolicy default
ErrorPolicy retry-job
</Printer>
```

Listing 18.11 »printers.conf« mit Quota-Einträgen

Sie können die Quotas auch direkt in die Dateien *printers.conf* und *classes.conf* eintragen. Denken Sie aber daran, vorher den CUPS-Server anzuhalten.

18.5 CUPS über die Kommandozeile

CUPS lässt sich auch über die Kommandozeile ansteuern, um Druckaufträge zu formatieren. In diesem Abschnitt werden einige Kommandos für die Formatierung von Druckaufträgen angesprochen, aber auch Kommandos für die Verwaltung von Druckern unter CUPS.

18.5.1 Einstellen eines Standarddruckers

Im Moment gibt es noch keinen Standarddrucker im System. Einen Standarddrucker können Sie über die Kommandozeile einstellen. ein Web-Frontend brauchen Sie dazu nicht. Sie benötigen eine Übersicht, welche Drucker in Ihrem System vorhanden sind. Diese Information können Sie mit dem Kommando `lpstat` abfragen. Listing 18.12 zeigt einen Auszug der Drucker, die in den vorigen Abschnitten eingerichtet wurden:

```
root@printserver:/etc/cups# lpstat -p -d
Drucker HL-L3230CDW ist im Leerlauf. Aktiviert seit Mi 28 Dez 2022 14:00:44 CET
Keine systemvoreingestellten Ziele
```

Listing 18.12 Übersicht über alle Drucker im System

Alle Drucker werden hier mit ihrem Erstellungsdatum und dem gegenwärtigen Status angezeigt. Die letzte Zeile des Listings zeigt an, dass auf diesem System kein Standarddrucker eingerichtet wurde. Das soll an dieser Stelle nachgeholt werden. Alle Optionen, die das Drucken betreffen, werden mit dem Kommando `lpoptions` verwaltet. In Listing 18.13 sehen Sie die Einstellungen für den Drucker, der als Standard festgelegt werden soll. Mit dem Kommando `lpoptions` soll jetzt auch der Standarddrucker eingerichtet werden:

```
root@printserver:~# lpoptions -d HL-L3230CDW
copies=1 device-uri=lpd://BRWEC5C687B1A7B/BINARY_P1 \
```

```

finishings=3 job-cancel-after=10800 job-hold-until=no-hold job-priority=50 \
job-sheets=none,none marker-change-time=0 number-up=1 \
printer-commands=AutoConfigure,Clean,PrintSelfTestPage \
printer-info='Brother HL-L3230CDW series' printer-is-accepting-jobs=true \
printer-is-shared=true printer-is-temporary=false printer-location \
printer-make-and-model='Brother HL-L3230CDW CUPS' printer-state=3 \
printer-state-change-time=1672232444 \
printer-state-reasons=cups-missing-filter-warning printer-type=8392796 \
printer-uri-supported=ipp://localhost/printers/HL-L3230CDW

```

```

root@printserver:~# lpstat -p -d
Drucker HL-L3230CDW ist im Leerlauf. Aktiviert seit Mi 28 Dez 2022 14:00:44 CET
systemvoreingestelltes Ziel: HL-L3230CDW

```

Listing 18.13 Einrichten eines Standarddruckers

Wenn Sie den Standarddrucker ändern, wird Ihnen immer die momentane Konfiguration des Druckers angezeigt. Lassen Sie sich im Anschluss daran wieder alle Drucker im System anzeigen, sehen Sie, dass der von Ihnen ausgewählte Drucker als Standarddrucker gesetzt ist.

18.5.2 Optionen für einen Drucker verwalten

Jeder Drucker kennt verschiedene Optionen für die Ansteuerung, sei es für die Papiergröße oder für den Papierschacht, aus dem gedruckt werden soll. Alle diese Optionen lassen sich über die Kommandozeile verändern. Sie haben die Möglichkeit, bestimmte Optionen nur einmal zu verwenden, oder Sie können die Optionen auch als *Printer Instances* speichern und später eine Instanz bei einem Ausdruck aufrufen, um die gewählten Optionen immer wieder verwenden zu können.

Jeder Benutzer hat eigene Optionen

Jeder Benutzer eines Druckers kann eigene Optionen für Drucker festlegen. Alle Optionen eines Benutzers werden in der Datei `./cups/lpoptions` abgelegt.

Damit Sie wissen, welche Parameter Ihr Drucker für die einzelnen Optionen überhaupt unterstützt, können Sie Ihren Drucker mit dem Kommando `lpoptions` so abfragen, wie es in Listing 18.14 zu sehen ist:

```

root@printserver:~# lpoptions -p HL-L3230CDW -l
PageSize/Media Size: *A4 Letter Legal Executive A5 A5Rotated ...
Duplex/Two-Sided: DuplexTumble DuplexNoTumble *None
BRInputSlot/Paper Source: *AutoSelect Tray1 Manual

```



```
BRResolution/Print Quality: *600dpi 600x2400dpi
BRMonoColor/Color / Mono: *Auto FullColor Mono
[...]
```

Listing 18.14 Abfrage der Werte, die ein Drucker unterstützt

Die hier angezeigten Werte unterscheiden sich je nachdem, wie Ihr Drucker ausgestattet ist. Jetzt können Sie die Einstellungen für den Drucker mit dem Kommando `lpoptions` vornehmen oder die Optionen für einen Ausdruck an das Kommando `lp` übergeben.

Als Erstes sollen die Optionen anhand von direkten Ausdrucken mit `lpr`, wie in Listing 18.15 zu sehen, erklärt werden:

```
root@printserver:/# lp -o media=A4 -o orientation-requested=3 textdatei.txt
root@printserver:/# lpr -n 2 -P HL-L3230CDW -o media=A4 \
-o job-hold-until=13:55 buch.pdf
```

Listing 18.15 Druckoptionen mit »lpr«

In diesem Listing wurden zwei Druckaufträge mit unterschiedlichen Parametern abgeschickt. Die Parameter haben die folgende Bedeutung:

- ▶ `-#2`
Durch diese Option können Sie die Anzahl der Kopien eines Ausdrucks angeben. Sie müssen testen, ob Ihr Drucker diese Option versteht oder als Option ein `-n 2` erwartet.
- ▶ `-o media=A4`
Mit dieser Option legen Sie die Papiergröße fest, die verwendet werden soll. Die von Ihrem Drucker unterstützten Größen können Sie mit `lpoption` abrufen.
- ▶ `orientation-requested=3`
Sie können über diese Option die Druckausrichtung festlegen. Für diese Option gibt es vier verschiedene Möglichkeiten, die den Ausdruck entsprechend drehen. Die vier Möglichkeiten sind:
 - `-o orientation-requested=3` – *portrait*-orientiert
 - `-o orientation-requested=4` – *landscape*-orientiert (90 Grad gedreht)
 - `-o orientation-requested=5` – *seascape*-orientiert (270 Grad gedreht)
 - `-o orientation-requested=6` – *reverse-portrait*-orientiert (180 Grad gedreht)
- ▶ `-P printer-01`
Der Druckauftrag soll, abweichend vom Standarddrucker, auf dem Drucker `printer-01` ausgegeben werden.
- ▶ `-o job-hold-until=13:55`
Dieser Job wird erst zu einer bestimmten Zeit gedruckt.

Am Ende aller Optionen wird die Datei angegeben, die gedruckt werden soll. Neben den hier aufgeführten Optionen gibt es noch eine Menge Optionen, mit denen Sie Druckaufträge steuern können. Alle Optionen können Sie in der Online-Hilfe des Web-Frontends, oder in der Manpage zu `lp` nachlesen. Alle Optionen, die direkt mit `lp` verwendet werden, sind nur für diesen einen Druckauftrag gültig.

Wollen Sie bestimmte Optionen häufiger verwenden, können Sie mit dem Kommando `lpoptions` eine zusätzliche Instanz für einen Drucker definieren. Wenn Sie später mit diesen Optionen drucken wollen, wählen Sie einfach die entsprechende Instanz aus. Am Beispiel von Listing 18.16 zeigen wir, wie Sie eine Instanz für einen Drucker einrichten können:

```
root@printserver:/# lpoptions -p HL-L3230CDW/spiegel -o mirror -o media=Transparency
```

Listing 18.16 Einrichten einer neuen Instanz

Die hier gesetzten Optionen finden Sie in der eigenen `lpoptions`-Datei wieder. Hier wird eine Instanz namens `spiegel` definiert. Bei dieser Instanz ist die Option `-o mirror` gesetzt, die einen Ausdruck spiegelverkehrt druckt. Zusätzlich wurde noch das Medium auf `Transparency` umgestellt. Immer wenn Sie nun diese Instanz verwenden, wird der Ausdruck spiegelverkehrt ausgedruckt und der Drucker an die Ausgabe auf Folie angepasst. Bevor wir die Instanz verwenden, zeigen wir Ihnen in Listing 18.17 noch mal einen Blick in die Datei `lpoptions`:

```
root@printserver:~# cat .cups/lpoptions
Default HL-L3230CDW
Dest HL-L3230CDW/spiegel media=Transparency mirror=true
```

Listing 18.17 Auszug aus der Datei »`lpoptions`« mit neuer Instanz

Im nächsten Schritt wird die neue Instanz getestet:

```
root@printserver:/etc/cups# lpr -P Farblaser/spiegel folien.pdf
```

Listing 18.18 Ausdruck mit der neuen Instanz

Jetzt wird als Drucker beim Ausdruck nicht nur der Name des Druckers angegeben, sondern zusätzlich noch die Instanz.

18.6 PPD-Dateien

Zum Abschluss werfen wir noch einen Blick auf die *Postscript Printer Description*-(PPD-)Dateien. Wenn Sie einen neuen Drucker in Ihr System einbinden, bekommen Sie heute fast immer mit dem Drucker auch die entsprechenden PPD-Dateien mitgeliefert.

Die PPD-Datei ist eine Textdatei, die Informationen über den Drucker bereitstellt. Zusätzlich enthält die Datei gerätespezifische PostScript-Formatierungen, die bestimmte Merkmale wie Ausgabemodus, Seitengrößen, eingebaute Fonts und andere benutzerdefinierbare

Eigenschaften festlegen. PPD-Dateien werden von CUPS wie Druckertreiber oder Spooler benutzt, um eine Bedienoberfläche für den Druckerdialog zu erzeugen und um Benutzereinstellungen und gerätespezifische Optionen zur Verfügung zu stellen.

In der Datei wird auch der von Ihnen bei der Einrichtung des Druckers ausgewählte Filter eingetragen, über den die Anpassung für die verwendete Sprache des Druckers durchgeführt wird. Bei allen Distributionen, die hier besprochen werden, sind direkt nach der Installation von CUPS PPD-Dateien für die meisten Drucker vorhanden, sodass es heute kein Problem mehr sein sollte, einen bestimmten Drucker in das System einzubinden.

18.7 Noch mehr Druck

Natürlich ist das bei Weitem noch nicht alles, was ein CUPS-Server Ihnen bieten kann. Es gibt viele weitere Optionen, die hier nicht besprochen wurden, und Sie haben auch noch die Möglichkeit, eine Benutzerauthentifizierung über verschiedene Quellen einzurichten.

Eine sehr gute Quelle für weitere Informationen ist immer die Online-Hilfe, die bei CUPS im Web-Frontend mitgeliefert wird. Der Vorteil ist hier, dass die Hilfe auch genau zu der installierten Version passt und Sie somit die Funktionen der auf Ihrem System installierten Version verwenden können.

TEIL IV
Infrastruktur

Kapitel 19

Hochverfügbarkeit

In diesem Kapitel lernen Sie, einen Aktiv/Passiv-Cluster mit hochverfügbarem Plattenplatz zu erstellen und zu betreiben.

Selbst dann, wenn Sie hochwertige Hardware kaufen und Ihre Dienste sorgfältig administrieren und überwachen, müssen Sie mit gelegentlichen Unterbrechungen leben. Hardware fällt aus, Server antworten wegen Überlast nicht, jemand zieht versehentlich das falsche Kabel aus dem Switch: All dies mindert die Verfügbarkeit.

Wenn Sie auf möglichst hohe Verfügbarkeit angewiesen sind – etwa weil auf Ihrem Webserver der Online-Shop läuft, mit dem Sie Geld verdienen –, werden Sie früher oder später über einen *Hochverfügbarkeits-Cluster* nachdenken. Ein Cluster besteht aus mindestens zwei Servern (den *Clusterknoten* oder *Nodes*) und verfügt über eine spezielle Software, die den Ausfall eines Dienstes oder eines ganzen Knotens erkennen kann. Als Reaktion auf den Ausfall startet die Software die ausgefallenen Dienste auf einem der verbleibenden Knoten neu.

Ein Hochverfügbarkeits-Cluster erleichtert auch Wartungsarbeiten. Ein Knoten kann kontrolliert aus dem Cluster herausgenommen und nach Abschluss der Arbeiten wieder integriert werden.

19.1 Das Beispiel-Setup

Sie erfahren in diesem Kapitel Schritt für Schritt, wie Sie einen Hochverfügbarkeits-Cluster installieren und in Betrieb nehmen. Als Lernbeispiel dient dabei ein praxisnaher Cluster:

- ▶ zwei Clusterknoten (*Node1* mit der IP-Adresse 10.0.0.1 und *Node2* mit der IP-Adresse 10.0.0.2)
- ▶ Aktiv/Passiv-Modell: Ein Knoten arbeitet, der zweite springt im Fehler- oder Wartungsfall ein.
- ▶ drei *Ressourcen*¹: eine IP-Adresse (10.0.0.10), der Webserver Apache und ein hochverfügbarer Plattenbereich, auf dem die Nutzdaten des Webserver liegen

¹ *Ressourcen* heißen im Clusterjargon alle Funktionen und Dienste, die im Fehlerfall auf einen anderen Knoten »umziehen«.

19.2 Installation

Je nachdem, welche Distribution Sie gewählt haben, gibt es kleine Unterschiede bei den Paketen, die Sie installieren müssen.

19.2.1 Debian 11 und Ubuntu 22.04 LTS

Alle benötigten Pakete sind bei Debian und Ubuntu in den Paketquellen enthalten. Führen Sie daher einfach den Befehl aus Listing 19.1 aus:

```
root@ubuntu:~# apt install drbd-utils pacemaker corosync crmsh resource-agents
```

Listing 19.1 Benötigte Pakete installieren

19.2.2 CentOS Stream

Einige der Pakete, die Sie benötigen, sind in CentOS nicht vorhanden. Sie können dieses Problem aber leicht lösen, indem Sie das *ELRepo* einbinden. Dieses Repository nennt sich selbst *The Community Enterprise Linux Repository*, und genau das stellt es dar: ein von der Community betriebenes Repository für Enterprise-Anforderungen. Zusätzlich müssen Sie (falls dies noch nicht geschehen ist) die HA-Repositories aktivieren. Um das *ELRepo* zu aktivieren und anschließend die Pakete zu installieren, genügen die Befehle aus Listing 19.2:

```
$ sudo rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
$ sudo dnf install -y https://www.elrepo.org/elrepo-release-9.el9.elrepo.noarch.rpm
$ sudo dnf config-manager --set-enabled highavailability
$ sudo dnf update
$ sudo dnf install drbd9x-utils kmod-drbd9x pacemaker corosync pcs
```

Listing 19.2 Hinzufügen des »ELRepo«



Fehler zur Drucklegung!

Leider bestand zu der Zeit, als dieses Buch in Druck ging, ein Fehler im *ELRepo*, wodurch das DRBD nicht lauffähig war – wir hoffen, dass dieser Fehler zeitnah korrigiert wird.

Standardmäßig lädt CentOS das Kernelmodul von DRBD nicht automatisch zur Laufzeit. Daher müssen wir dies (einmalig) mit dem Befehl `sudo modprobe drbd` nachholen. Zusätzlich müssen Sie noch folgende Befehle absetzen, damit die Kommunikation zwischen den Systemen erlaubt wird:

```
$ sudo firewall-cmd --permanent --add-service=high-availability
$ sudo firewall-cmd --reload
```

Listing 19.3 Firewall-Anpassungen

19.2.3 openSUSE Leap

Installieren Sie zunächst alle benötigten Pakete mit dem folgenden Kommando:

```
daniel@leap:~> sudo zypper install pacemaker corosync drbd crmsh
```

Listing 19.4 openSUSE Leap: benötigte Pakete installieren

Anschließend müssen Sie noch das Kernelmodul von DRBD laden und dafür sorgen, dass Pacemaker standardmäßig beim Systemstart aktiviert wird:

```
daniel@leap:~> sudo modprobe drbd
daniel@leap:~> sudo sh -c "echo 'drbd' > /etc/modules-load.d/drbd.conf"
daniel@leap:~> sudo firewall-cmd --zone=trusted --add-source=10.0.0.0/24 --permanent
daniel@leap:~> sudo firewall-cmd --reload
```

Listing 19.5 Laden von DRBD und permanentes Aktivieren von Pacemaker

19.3 Einfache Vorarbeiten

Sorgen Sie unbedingt dafür, dass jeweils beide Clusterknoten in der Datei `/etc/hosts` eingetragen sind. Sie riskieren sonst schwer zu lokalisierende Fehler, wenn der Nameserver einmal nicht erreichbar sein sollte. In unserem Szenario mit den beiden Knoten `Node1` und `Node2` sieht die Datei so aus:

```
10.0.0.1      node1.example.com  node1
10.0.0.2      node2.example.com  node2
```

Listing 19.6 Ergänzung in der »`/etc/hosts`« auf beiden Nodes

Außerdem ist es wichtig, dass Ihre beiden Knoten sich einig sind, was die aktuelle Uhrzeit angeht. Um dies sicherzustellen, installieren Sie am besten einen *Network Time Protocol Daemon*. Bei Debian und Ubuntu wird standardmäßig bereits der Zeitgeber von `systemd` (`timesyncd`) installiert und gestartet. Unter openSUSE Leap und CentOS wird dafür `chrony` eingesetzt. Der Dienst ist nach der Installation bereits aktiv. Um zu überprüfen, ob Ihr System die Zeit via NTP bezieht, können Sie einfach den Befehl `timedatectl status` absetzen. Dieser ermittelt über den `systemd-timesyncd` die Konfiguration Ihres Systems (egal ob es mit `timesyncd` oder `chrony` arbeitet) und gibt die Konfiguration aus.

19.4 Shared Storage mit DRBD

DRBD steht für *Distributed Replicated Block Device* und wird gern als *RAID-1 übers Netz* bezeichnet. Jeder der beiden Clusterknoten besitzt eine DRBD-Partition, die von einem speziellen Block-Device repräsentiert wird, etwa `/dev/drbd0`. Von diesen Block-Devices ist eines

aktiv (*primary*) und das andere passiv (*secondary*). Alle Daten, die auf die aktive Partition geschrieben werden, spiegelt DRBD über das Netz auf die passive Partition. Ein solcher Replikationsverbund wird als *DRBD-Ressource* bezeichnet. Konfiguriert wird DRBD in der Konfigurationsdatei */etc/drbd.conf*. Diese lädt bzw. inkludiert aber nur die Dateien unterhalb von */etc/drbd.d/*. Daher legen wir die Konfiguration auch dort ab.

19.4.1 Grundlegende Konfiguration

In der Konfigurationsdatei */etc/drbd.d/global_common.conf* finden Sie, wie ihr Name schon andeutet, die Abschnitte *global* und *common*:

```
global {
    usage-count yes;    ### steht standardmäßig auf 'no'
}
common {
    [...]
}
```

Listing 19.7 Die Konfigurationsdatei »*global_common.conf*«

Für jede Ressource, die Sie benutzen möchten, legen Sie eine Datei unter */etc/drbd.d/* an. Der Dateiname muss auf *.res* enden. Ansonsten sind Sie in der Namensgebung frei. Es empfiehlt sich aber, den Namen der definierten Ressource auch als Dateinamen zu benutzen. In diesem Fall verwenden wir *mydrbd.res*:

```
resource mydrbd {
    net {
        cram-hmac-alg sha1;
        shared-secret "geheim";
    }
    on node1 {
        device /dev/drbd0;
        disk /dev/sdb;
        address 10.0.0.1:7789;
        meta-disk internal;
    }
    on node2 {
        device /dev/drbd0;
        disk /dev/sdb;
        address 10.0.0.2:7789;
        meta-disk internal;
    }
}
```

Listing 19.8 Die Konfigurationsdatei »*mydrbd.res*«



Die Konfigurationsdateien müssen auf beiden Clusterknoten identisch sein. Kopieren Sie diese daher am besten mit Copy-and-paste oder übertragen Sie die Dateien via SSH.

19.4.2 Die wichtigsten Konfigurationsoptionen

In den Konfigurationsdateien aus unserem Beispiel sind alle zum Betrieb einer DRBD-Ressource benötigten Parameter enthalten, auch wenn sich natürlich noch mehr konfigurieren ließe. Diese Parameter haben folgende Bedeutung:

- ▶ `usage-count yes`
Die DRBD-Entwickler möchten gern wissen, wie oft ihre Software installiert wurde. Wenn diese Option auf `yes` gesetzt ist, nimmt Ihre DRBD-Installation an der Zählung teil. Beim ersten Start erhalten Sie folgende Meldung:

```

---== Thank you for participating in the global usage survey ===-
The server's response is:

you are the 3948th user to install this version

```

Listing 19.9 DRBD-Online-Umfrage

- ▶ `resource mydrbd`
An dieser Stelle vergeben Sie den Namen für Ihre DRBD-Ressource.
- ▶ `cram-hmac-alg sha1; shared-secret "geheim"`
Dieses `shared secret` benutzen die Knoten, um sich gegenseitig zu authentifizieren. Falls Sie mehrere Ressourcen verwalten, sollten Sie für jede Ressource ein anderes `shared secret` nutzen. Sie verhindern so, dass – vielleicht durch einen Tipp- oder Kopierfehler – eine neue Ressource einer bereits bestehenden in die Kommunikation pfuscht.
- ▶ `on <HOSTNAME>`
Diese Zeile leitet die knotenspezifische Konfiguration ein. Hier wird der Kurzname des Knotens verwendet, wie er in der `/etc/hosts` eingetragen ist. Bei CentOS wird standardmäßig als Hostname der FQDN verwendet. Achten Sie daher darauf, hier den korrekten Wert anzugeben – am einfachsten ermitteln Sie diesen mit dem Programm `hostname`.
- ▶ `device`
Der Name des DRBD-Block-Device. Sie verwenden ihn, wie Sie in den folgenden Beispielen sehen werden, für Dateisystemoperationen wie etwa das Formatieren.
- ▶ `disk`
Hier wird die Plattenpartition definiert, auf die DRBD physisch zugreift. DRBD übernimmt die Verwaltung dieser Partition vollständig. Sie als Admin arbeiten nur noch mit dem `device (/dev/drbd0)` oder dem Ressourcennamen (`mydrbd`) und führen keine Low-Level-Operationen auf der `disk` aus.

- ▶ `address`
Dies ist die IP-Adresse des Knotens und der Port, über den die Kommunikation mit dem Nachbarknoten abgewickelt wird.
- ▶ `meta-disk`
Darüber wird festgelegt, dass das DRBD-Block-Device eine reguläre Partition ist und nicht etwa von LVM verwaltet wird, da hierfür weitere Konfigurationen notwendig wären.

19.4.3 Die DRBD-Ressource in Betrieb nehmen

Bevor Sie Ihre DRBD-Ressource in Betrieb nehmen, sollten Sie die nachstehende Checkliste durchgehen, damit Ihnen die DRBD-Konfiguration auf Anhieb gelingt:

1. Prüfen Sie noch einmal, ob die Konfigurationsdateien auf beiden Nodes identisch sind.
2. Die Plattenpartitionen, die Sie für DRBD nutzen, dürfen nicht formatiert sein – sie müssen aber partitioniert sein! Prüfen Sie auf beiden Knoten, ob dort vielleicht noch ein Filesystem angelegt ist. In diesem Fall können Sie es so zerstören:²

```
dd if=/dev/zero of=/dev/sdb bs=1M count=128
```
3. Führen Sie auf beiden Nodes den Befehl `drbdadm create-md mydrbd` als root-Benutzer aus.
4. Starten Sie mit `systemctl restart drbd` auf beiden Knoten den DRBD-Dienst neu.
5. Lassen Sie sich nun mit dem Befehl `cat /proc/drbd` den DRBD-Status anzeigen.

Immer den Überblick behalten

Lassen Sie sich in zwei separaten Terminals immer die aktuelle Ausgabe von `drbdadm status` beider Knoten anzeigen. Durch Eingabe von `watch "cat /proc/drbd"` wird der Befehl alle zwei Sekunden ausgeführt und das Ergebnis angezeigt – so behalten Sie immer den Überblick, in welchem Status sich Ihre DRBD-Devices befinden.

CentOS und openSUSE Leap: Fehler zur Drucklegung

Leider wird unter diesen Distributionen `/proc/drbd` nicht korrekt befüllt. Nutzen Sie dafür alternativ die Ausgabe von `drbdadm status`.

openSUSE: Fehler zur Drucklegung

Leider wird `drbd` mit einem fehlerhaften `systemd`-Skript ausgeliefert. Falls Ihre Instanz nicht starten möchte, können Sie den Fehler so korrigieren:

```
$ sudo cp /usr/lib/systemd/system/drbd.service /etc/systemd/system/  
$ sudo sed -i 's#/lib/drbd/#usr/lib/drbd/#g' /etc/systemd/system/drbd.service  
$ sudo systemctl daemon-reload
```

² An dieser Stelle sei ein Hinweis auf Kapitel 7, »Backup und Recovery«, gestattet.

Die Ausgabe sieht in dieser Phase auf beiden Knoten so aus:

```
root@node2:~# cat /proc/drbd
version: 8.4.11 (api:1/proto:86-101)
srcversion: FC343D849E3B88C1E7B55C
 0: cs:Connected ro:Secondary/Secondary ds:Inconsistent/Inconsistent C r-----
    ns:0 nr:0 dw:0 dr:0 al:8 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:8386268
```

Listing 19.10 Ausgabe von »cat /proc/drbd«

Secondary/Secondary bedeutet, dass Sie noch keiner der beiden Seiten die aktive Rolle (*primary*) zugewiesen haben. Inconsistent heißt, dass die Knoten noch nicht synchronisiert wurden und somit noch nicht den gleichen Datenbestand haben. Um zu testen, ob sich die Knoten miteinander synchronisieren lassen, geben Sie auf *Node1* die folgenden Befehle ein:

```
$ drbdadm up mydrbd
$ drbdadm primary mydrbd --force
```

Listing 19.11 Knoten miteinander synchronisieren, *Node1* ist »Primary«.

Dieser Befehl synchronisiert die beiden DRBD-Devices miteinander, wobei *Node1* die Primary-Rolle zugewiesen wird. Sein Datenbestand wird also auf *Node2* übertragen.

Da beide Seiten noch kein Filesystem haben, wird hier nur Datenschnitt synchronisiert, aber für einen Funktionstest ist das ausreichend. Während der Synchronisation sieht cat /proc/drbd auf *Node1* so aus:

```
[...]
0: cs:SyncSource ro:Primary/Secondary ds:UpToDate/Inconsistent C r-----
  ns:5041200 nr:0 dw:69864 dr:4974588 al:14 bm:0 lo:0 pe:2 ua:0 ap:0 ep:1 wo:f \
  oos:3353200
  [=====>.....] sync'ed: 60.1% (3272/8188)M
  finish: 0:01:25 speed: 39,420 (29,968) K/sec
```

Listing 19.12 *Node1* wird »Primary« und überschreibt Daten auf *Node2*.

Sie sehen, dass *Node1* nun den Primary-Status hat. Der Zustand des Datenbestands ist UpToDate für *Node1* und Inconsistent für *Node2*, da die Synchronisation noch nicht abgeschlossen ist. Die Synchronisierung kann je nach Größe der Partition und Geschwindigkeit der Datenverbindung zwischen den beiden Knoten eine Zeit lang dauern. Ist der Vorgang abgeschlossen, sieht die Ausgabe von cat /proc/drbd wie folgt aus:

```
[...]
0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r-----
  ns:8392444 nr:0 dw:69864 dr:8325832 al:14 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:0
```

Listing 19.13 Synchronisation abgeschlossen, *Node1* ist »Primary«.

Der Datenbestand auf beiden Nodes hat jetzt den Zustand `UpToDate`. Falls die Synchronisation abgeschlossen ist, aber immer noch `Secondary/Secondary` angezeigt wird, können Sie *Node1* mit dem folgenden Befehl in den Primary-Status versetzen:

```
$ drbdadm primary mydrbd --force
```

Listing 19.14 Primary-Status erzwingen

Nachdem der Funktionstest erfolgreich verlaufen ist, können Sie das Device jetzt formatieren. Legen Sie auf *Node1* mit dem folgenden Befehl ein Ext4-Dateisystem an:

```
$ mkfs.ext4 /dev/drbd0
```

Listing 19.15 Dateisystem anlegen

Es muss natürlich nicht unbedingt Ext4 sein: Jedes andere gängige Dateisystem funktioniert ebenfalls. Sie können */dev/drbd0* nun mounten. Da auf dem DRBD-Device später die Daten eines Webangebots liegen sollen, nennen Sie den zugehörigen Mountpunkt */srv/www-data*:

```
# fuehren Sie diesen Befehl als Root-Benutzer auf beiden Knoten aus:  
mkdir /srv/www-data
```

```
# fuehren Sie diesen Befehl nur auf Node1 aus:  
mount /dev/drbd0 /srv/www-data
```

Listing 19.16 Mountpunkt erstellen und DRBD-Device einbinden



Beachten Sie, dass Sie ein DRBD-Device nur auf dem Knoten mounten können, auf dem es den Status `primary` hat.

Damit Sie später, wenn Sie das Hochverfügbarkeits-Setup testen, etwas Inhalt auf dem Dateisystem haben, legen Sie nun dort eine kleine HTML-Datei ab. Zum Testen reicht schon die klassische Ausgabe von »Hello World«:

```
<html>  
<head>  
  <title>HA-Test</title>  
</head>  
<body>  
  <h1>Hello World!</h1>  
  Dies ist eine Testseite.  
</body>  
</html>
```

Listing 19.17 Einfache HTML-Datei

Legen Sie die Datei unter dem Namen *index.html* im Verzeichnis */srv/www-data* ab. Damit ist die Grundkonfiguration von DRBD abgeschlossen. Unmounten Sie das Filesystem wie-

der, denn Mounts und Unmounts sollen bald vom Clusterressourcenmanager übernommen werden, dessen Konfiguration als Nächstes an der Reihe ist.

19.5 Grundkonfiguration der Clusterkomponenten

Um das Hochverfügbarkeits-Setup in Betrieb nehmen zu können, müssen Sie noch zwei Komponenten konfigurieren. Die erste Komponente ist ein Benachrichtigungssystem, mit dem die einzelnen Knoten eines Clusters die Möglichkeit haben, Nachrichten über ihren aktuellen Status untereinander auszutauschen und festzustellen, ob einer der Knoten eventuell ausgefallen ist und nicht mehr antwortet. In diesem Fall müssen die Ressourcen, die der ausgefallene Knoten bereitgestellt hat, auf einem anderen Knoten neu gestartet werden. Diese Arbeit übernimmt die zweite Komponente, der *Cluster Resource Manager*.

19.5.1 Pacemaker und Corosync: das Benachrichtigungssystem

Die Kontrolle der Lebenszeichen und die Abwicklung des Nachrichtenverkehrs zwischen den Clusterknoten übernimmt *Corosync* in Zusammenarbeit mit *Pacemaker*.

Die Konfigurationsdatei anpassen

Die Datei *corosync.conf* liegt auf Debian- und Ubuntu-Systemen unter */etc/corosync/*. Unter openSUSE Leap muss zunächst die Beispieldatei *corosync.conf.example*, die unter */etc/corosync/* zu finden ist, nach *corosync.conf* kopiert werden. Unter CentOS wird die Konfigurationsdatei mit dem Programm *pcs* erzeugt – als CentOS-Nutzer überspringen Sie daher den folgenden Teil, und fahren ab »Vorbereitungen für CentOS« weiter fort. Passen Sie den vorhandenen Konfigurationsblock *interface* an oder legen Sie diesen mit den Werten aus Listing 19.18 an. Ändern Sie den Wert hinter *bindnetaddr* in die IP-Adresse des Clusterknotens, auf dem Sie sich gerade befinden. Der Block sieht danach so aus:

```
interface {
    ringnumber: 0
    bindnetaddr: 10.0.0.1
    mcastaddr: 239.255.1.1
    mcastport: 5405
    ttl: 1
}
```

Listing 19.18 Interfacekonfiguration für das Clustermessaging

Die vorgegebene Multicast-Adresse *mcastaddr: 239.255.1.1* muss auf allen Clusterknoten, die in einem Verbund arbeiten sollen, gleich sein. Wenn Sie mehrere Cluster betreiben, vergeben Sie für jeden Cluster eine andere Multicast-Adresse.



Anschließend müssen Sie im Block `nodelist` jeweils einen `node`-Block für jeden Clusterknoten hinzufügen. Der gesamte Block sieht danach wie folgt aus:

```
nodelist {
  node {
    name: node1
    nodeid: 1
    ring0_addr: 10.0.0.1
  }
  node {
    name: node2
    nodeid: 2
    ring0_addr: 10.0.0.2
  }
}
```

Listing 19.19 Node-Konfiguration für das Clustermessaging

Schließen Sie nun die Konfigurationsdatei auf diesem Knoten, und passen Sie die Konfigurationsdatei auf dem zweiten Knoten nach dem gleichen Muster an. Mit Ausnahme der `bindnetaddr`-Einstellung, in der die IP-Adresse des jeweiligen Knotens steht, soll die Datei auf beiden Knoten identisch sein.

Vorbereitungen für CentOS

Leider arbeitet CentOS elementar anders als die übrigen Distributionen. Daher müssen Sie, bevor wir das Programm `pcs` einsetzen können, noch einen Cluster erzeugen. Führen Sie dafür auf beiden Knoten die folgenden Befehle aus, jeweils einen nach dem anderen:

```
$ systemctl enable pcsd
$ systemctl start pcsd
$ sudo passwd hacluster
$ sudo pcs host auth node1
Username: hacluster
Password: <PASSWORD>
node1: Authorized
$ sudo pcs host auth node2
Username: hacluster
Password: <PASSWORD>
node2: Authorized
```

Listing 19.20 Die Authentifizierung eines Clusters unter »CentOS« einrichten

Mit diesen Befehlen aktivieren und starten Sie den Cluster und setzen das Passwort für den Benutzer `hacluster`, der zentral zur Authentifizierung der Nodes verwendet wird. Bei dem

Kommando `pcs host auth <NODE>` werden Sie nach der Authentifizierung für diesen Node gefragt – geben Sie hier das soeben vergebene Passwort des hacluster-Benutzers an.

Anschließend müssen Sie noch einen Cluster anlegen und zwei Konfigurationen (da wir nur zwei Nodes betreiben) absetzen – diese Befehle müssen nur auf einem Node ausgeführt werden:

```
$ sudo pcs cluster setup --start MyCluster node1 node2
No addresses specified for host 'node1', using 'node1'
No addresses specified for host 'node2', using 'node2'
Destroying cluster on hosts: 'node1', 'node2'...
node1: Successfully destroyed cluster
node2: Successfully destroyed cluster
Requesting remove 'pcsd settings' from 'node1', 'node2'
node1: successful removal of the file 'pcsd settings'
node2: successful removal of the file 'pcsd settings'
Sending 'corosync authkey', 'pacemaker authkey' to 'node1', 'node2'
node1: successful distribution of the file 'corosync authkey'
node1: successful distribution of the file 'pacemaker authkey'
node2: successful distribution of the file 'corosync authkey'
node2: successful distribution of the file 'pacemaker authkey'
Sending 'corosync.conf' to 'node1', 'node2'
node1: successful distribution of the file 'corosync.conf'
node2: successful distribution of the file 'corosync.conf'
Cluster has been successfully set up.
Starting cluster on hosts: 'node1', 'node2'...
```

Listing 19.21 Erstellen und Konfigurieren eines Clusters unter »CentOS«

Wie Sie Listing 19.21 entnehmen können, wird zunächst ein möglicher vorhandener Cluster gelöscht und anschließend neu erstellt. Dabei wird auch der sogenannte authkey erzeugt. Mit diesem Schlüssel weisen sich die Knoten untereinander aus. Der erzeugte Schlüssel liegt im Übrigen im Corosync-Konfigurationsverzeichnis.

Mit dem nachstehenden Befehl können Sie ihn neu erzeugen:

```
$ corosync-keygen
```

Listing 19.22 »authkey« erzeugen

Zu langsam? Installieren Sie »haveged«!

Falls die Erstellung eines Keys ewig benötigt, verfügt Ihr System über zu wenig *Entropie*. Installieren Sie einfach das Paket *haveged*, und schon müssen Sie beim Erzeugen von Keys keine Kaffeepause mehr einlegen.



Diensteinrichtung

Die Konfiguration des Benachrichtigungssystems ist damit abgeschlossen, es kann gestartet werden. Der Cluster wurde direkt nach der Konfiguration mittels `systemctl restart corosync` neu gestartet und hat direkt seinen Betrieb aufgenommen. Zusätzlich sollten Sie prüfen, ob der Ressourcenmanager *Pacemaker* bereits gestartet ist. Falls nicht, führen Sie `systemctl restart pacemaker` aus, um dies nachzuholen.

Da der Cluster diese Dienste stets benötigt, sollten Sie auch sicherstellen, dass sie beim Systemstart automatisch gestartet werden. Führen Sie daher zur Sicherheit die Befehle aus Listing 19.23 aus:

```
$ systemctl enable corosync && systemctl start corosync
$ systemctl enable pacemaker && systemctl start pacemaker
```

Listing 19.23 Dienste automatisch beim Systemstart starten

19.5.2 Pacemaker: der Ressourcenmanager

Unter Debian, Ubuntu und openSUSE Leap erfolgt die Konfiguration von Pacemaker über das Programm `crm`. Unter CentOS wurde dieses Programm durch `pcs` ersetzt. Die Programme arbeiten ähnlich, aber nicht gleich – daher werden wir Ihnen die Unterschiede jeweils darstellen. Unter einer *Ressource* versteht man alles, was im Fehlerfall auf einen anderen Knoten umziehen muss: IP-Adressen, Dienste, DRBD-Devices.

Pacemaker verwaltet den Start, den Stopp und die kontrollierte Migration der Ressourcen und kann dabei auch Abhängigkeiten berücksichtigen.

Sie haben zwar noch keine Ressourcen konfiguriert, aber Ihre beiden Nodes kennt Pacemaker bereits. Geben Sie einmal auf der Kommandozeile den Befehl `crm status` ein (unter CentOS: `pcs status`). Sie erhalten eine solche Ausgabe:

```
root@node2:~# crm status                                     ### CentOS: pcs status
Cluster Summary:
* Stack: corosync
* Current DC: node1 (version 2.1.2-ada5c3b36e2) - partition with quorum
* Last updated: Fri Dec 30 08:41:34 2022
* Last change:  Fri Dec 30 09:40:04 2022 by hacluster via crmd on node1
* 2 nodes configured
* 0 resource instances configured

Node List:
* Online: [ node1 node2 ]
[...]
```

Listing 19.24 CRM-Status, Nodes ohne Ressourcen

Pacemaker: Den Status im Blick behalten

Der Befehl `crm_mon` erzeugt die gleiche Ausgabe wie `crm status`, allerdings wird die Ausgabe laufend aktualisiert. So können Sie immer verfolgen, welchen Status Ihre Nodes und Ressourcen gerade haben. Für CentOS mit `pcs` existiert diese Möglichkeit leider nicht.

Nodes online und offline schalten

Wenn Sie einen Node, beispielsweise wegen notwendiger Wartungsarbeiten, aus dem Cluster herausnehmen müssen, bietet Pacemaker Ihnen dafür eine einfache Möglichkeit.

Geben Sie auf dem betreffenden Node dieses Kommando ein:

```
# Debian/Ubuntu/openSUSE Leap
crm node standby
```

```
# CentOS
pcs node standby
```

Listing 19.25 Einen Knoten auf »Standby« schalten

Wenn Sie gerade nicht auf dem Node eingeloggt sind, den Sie auf Standby schalten wollen, können Sie die Aktion trotzdem ausführen. Geben Sie das Kommando auf einem anderen Clusterknoten ein, und hängen Sie den Namen des Knotens an, den Sie stoppen möchten, so wie in Listing 19.26 dargestellt:

```
# Debian/Ubuntu/openSUSE Leap
crm node standby node2
```

```
# CentOS
pcs node standby node2
```

Listing 19.26 Einen Knoten remote auf »Standby« schalten

Die Ausgabe von `crm status` zeigt die Zustandsänderung an:

```
root@node2:~# crm status      ### CentOS: pcs status
[...]
* 2 nodes configured
* 0 resource instances configured
```

Node List:

```
* Node node2: standby
* Online: [ node1 ]
```

Listing 19.27 CRM-Status, ein Node im Standby-Betrieb

Nach Abschluss der Arbeiten können Sie den Knoten mit dem folgenden Befehl wieder in den Cluster integrieren:

```
# Debian/Ubuntu/openSUSE Leap
crm node online
```

```
# CentOS
pcs node unstandby
```

Listing 19.28 Den Knoten wieder aktivieren

19.5.3 Ein Quorum deaktivieren

In der Ausgabe des Befehls `crm status` taucht unter anderem diese Zeile auf:

```
[...]
* Current DC: node2 (version 2.1.2-ada5c3b36e2) - partition with quorum
[...]
```

Listing 19.29 Quorum in der Statusausgabe

Ein Quorum ist ein Datenbestand, der für alle Knoten stets zugänglich ist und den aktuellen Status des Clusters enthält. Dazu zählt beispielsweise, welcher Knoten für eine bestimmte Ressource aktuell die *Master*-Rolle innehat, und vieles mehr.

Sollte es zu einer *Split-Brain*-Situation kommen (Teile des Clusters können nicht mehr miteinander kommunizieren und wissen nicht, ob ihre Nachbarknoten noch funktionieren oder nicht), kann anhand des Quorums sehr schnell entschieden werden, wie viele Knoten des Clusters deaktiviert werden und welche weiterarbeiten dürfen.

Unser Beispiel-Cluster hat aber nur zwei Knoten – in einer *Split-Brain*-Situation ist von Anfang an klar, dass bei einem Ausfall der Kommunikationsverbindung zwischen den Knoten nur einer von beiden weiterarbeiten darf, um Inkonsistenzen im Datenbestand zu verhindern.³ Ein Quorum ist deshalb in 2-Node-Clustern überflüssig. Sie können es mit diesem Befehl deaktivieren:

```
# Debian/Ubuntu/openSUSE Leap
crm configure property stonith-enabled=false
crm configure property no-quorum-policy=ignore
```

```
# CentOS (bereits bei den Vorbereitungen umgesetzt)
pcs property set stonith-enabled=false
pcs property set no-quorum-policy=ignore
```

Listing 19.30 Quorum deaktivieren

³ Siehe dazu auch Abschnitt 19.6.4, »Fencing«.

Die Pacemaker-Shell

Pacemaker bringt eine eigene Shell mit. Mit ihr sind alle CRM-Befehle, die als Kommandozeilenparameter funktionieren, auch über eine Menüstruktur verfügbar (auf CentOS mit pcs gibt es diese Möglichkeit leider nicht).

Ein Beispiel: Einen Node können Sie auf der Kommandozeile, wie Sie gerade gesehen haben, mit dem Befehl `crm node standby offline` schalten. Mit der CRM-Shell geht das wie folgt:

1. Geben Sie auf der Kommandozeile `crm` ohne weitere Parameter ein. Der Prompt ändert sich in `crm(live/<NODE NAME>)#`.
2. Um einen Überblick über Ihre Möglichkeiten zu bekommen, tippen Sie `help`. Sie erhalten diese Ausgabe:

```
[...]
Available commands:

      cd          Navigate the level structure
      help        Show help (help topics for list of topics)
      ls          List levels and commands
      quit        Exit the interactive shell
      [...]
```

Listing 19.31 Auszug der verfügbaren Kommandos oder Untermenüs

Zum Teil sind dies Befehle, die Sie ausführen können (status kennen Sie bereits), zum Teil können Sie auch tiefer in die Menüstruktur einsteigen.

3. Geben Sie auf der Kommandozeile einmal `node` ein. Sie befinden sich jetzt in einem Untermenü, in dem Sie die Knoten Ihres Clusters verwalten können. Der Kommandozeilenprompt hat sich in `crm(live/<NODE NAME>)node#` geändert. Wieder zeigt `help` Ihnen an, welche Möglichkeiten Sie haben:

```
Node management and status commands.

Commands:
      attribute   Manage attributes
      clearstate  Clear node state
      delete      Delete node
      fence       Fence node
      maintenance Put node into maintenance mode
      online      Set node online
      ready       Put node into ready mode
      server      Show node hostname or server address
      show        Show node
      standby     Put node into standby
      status      Show nodes' status as XML
```

status-attr	Manage status attributes
utilization	Manage utilization attributes
cd	Navigate the level structure
help	Show help (help topics for list of topics)
ls	List levels and commands
quit	Exit the interactive shell
up	Go back to previous level

Listing 19.32 Verfügbare Kommandos im »node«-Untermenü

Hier finden Sie unter anderem die Kommandos `standby` und `online` wieder, die Sie auf der Kommandozeile bereits benutzt haben.

4. Durch Eingabe von `standby` oder `online` können Sie den aktuellen Knoten stoppen und wieder starten. Mit `quit` verlassen Sie die CRM-Shell, und mit `up` gelangen Sie in die nächsthöhere Menüebene zurück.

19.6 Dienste hochverfügbar machen

Jetzt können Sie darangehen, eine erste Ressource zu konfigurieren. Die Auswahl ist groß: Pacemaker kann auf vorgefertigte Routinen für fast alle gängigen Ressourcen zurückgreifen. In den nächsten Abschnitten werden Sie sehen, wie Sie diese Routinen nutzen können. Wenn Sie neugierig sind, welche Ressourcen Ihnen zur Verfügung stehen, geben Sie einmal folgendes Kommando ein:

```
crm ra list ocf
```

Listing 19.33 Ressourcen-Namespaces anzeigen

Sie erhalten eine Liste von OCF-Ressourcen (unter CentOS mit `pcs` liefert der Befehl `pcs resource agents` eine ähnliche Ausgabe), die Sie konfigurieren und einsetzen können (OCF steht dabei für *Open Cluster Framework*, ein Standardisierungsregelwerk für Clusterressourcen):

AoEtarget	AudibleAlarm	CTDB
ClusterMon	Delay	Dummy
EvmsSCC	Evmsd	Filesystem
HealthCPU	HealthIOWait	HealthSMART
ICP	IPaddr	IPaddr2
IPsrcaddr	IPv6addr	LVM
LVM-activate	LinuxSCSI	MailTo
ManageRAID	ManageVE	NodeUtilization
Pure-FTPd	Raid1	Route
SAPDatabase	SAPInstance	SendArp

ServerAID	SphinxSearchDaemon	Squid
Stateful	SysInfo	SystemHealth
VIPArIp	VirtualDomain	WAS
WAS6	WinPopup	Xen
[...]		

Listing 19.34 Verfügbare OCF-Ressourcen

Als erste Ressource benötigen Sie eine IP-Adresse, an die Sie später einen hochverfügbaren Dienst binden können.

19.6.1 Die erste Ressource: eine hochverfügbare IP-Adresse

Da die IP-Adresse, die wir hier als Ressource definieren wollen, zwischen den Knoten »wandern« kann, darf sie natürlich nicht auf einem der beiden Knoten bereits in Benutzung sein. Für das Beispiel muss die IP-Adresse 10.0.0.10 erhalten.

Da wir ja schon über einen lauffähigen Cluster verfügen, genügt es im Übrigen, diese Konfiguration auf einem Knoten durchzuführen. So konfigurieren Sie die Ressource auf der Kommandozeile:

```
### Debian/Ubuntu/openSUSE Leap
crm configure primitive myMIP ocf:heartbeat:IPaddr2 params ip=10.0.0.10 \
cidr_netmask=32

### CentOS
pcs resource create myMIP ocf:heartbeat:IPaddr2 ip=10.0.0.10 cidr_netmask=32
```

Listing 19.35 IP-Adresse als Ressource hinzufügen

Geben Sie den Befehl auf einer Zeile ohne den Backslash ein. Der Name der Ressource ist frei wählbar, solange Sie auf Umlaute und Leerzeichen verzichten. Das »MIP« in myMIP steht hier für *Migrating IP*, also »wandernde IP-Adresse«.

In der Ausgabe von `crm status` sehen Sie, dass die Adresse bereits aktiv ist:

```
[...]
* 2 nodes configured
* 1 resource instance configured

Node List:
* Online: [ node1 node2 ]

Full List of Resources:
* myMIP (ocf:heartbeat:IPaddr2): Started node1
```

Listing 19.36 Die IP-Ressource ist aktiv.

Hier können Sie auch erkennen, auf welchem Clusterknoten die IP-Adresse gerade beheimatet ist: Started node1.

Wenn Sie diesen Knoten jetzt auf Standby schalten, migriert die Ressource automatisch auf den zweiten Knoten:

```
### Debian/Ubuntu/openSUSE Leap
root@node2:~# crm node standby node1 && sleep 1 && crm status

### CentOS
[root@node2 ~]# pcs node standby node1 && sleep 1 && pcs status

### Ausgabe:
[...]
Node List:
  * Node node1: standby
  * Online: [ node2 ]
```

```
Full List of Resources:
  * myMIP (ocf:heartbeat:IPaddr2): Started node2
```

Listing 19.37 Knoten im Standby: Die Ressource migriert.

Die Ressource starten, stoppen und migrieren

Wenn Sie eine Ressource zeitweilig deaktivieren möchten, lautet der passende Befehl `crm resource stop <RESOURCE>`⁴.

```
### Ressource myMIP stoppen ...
# Debian/Ubuntu/openSUSE Leap
crm resource stop myMIP

# CentOS
pcs resource disable myMIP

### ... und wieder starten
#Debian/Ubuntu/openSUSE Leap
crm resource start myMIP

# CentOS
pcs resource enable myMIP
```

Listing 19.38 Eine Ressource stoppen und wieder starten

⁴ Das englische Wort *resource* hat, anders als sein deutsches Äquivalent, nur ein *s* – der Tippfehler Nummer eins beim Clusterbau.

Sie können eine Ressource anweisen, auf einen anderen Knoten umzuziehen. Am Beispiel der Ressource *myMIP* lautet das Kommando dazu:

```
# Debian/Ubuntu/openSUSE Leap
crm_resource --resource myMIP --move
```

```
# CentOS
pcs resource move myMIP
```

Listing 19.39 Die Ressource »myMIP« zur Migration zwingen

Migration ohne Wiederkehr

Der Knoten, auf dem die Ressource bisher lief, ist nach der Zwangsmigration für sie tabu. Die Ressource würde nicht einmal auf diesen Knoten zurückmigrieren, wenn er der letzte verbleibende Node im Cluster wäre.

Um die Rückmigration wieder zu erlauben, geben Sie das folgende Kommando ein:

```
# Debian/Ubuntu/openSUSE Leap
crm_resource --resource myMIP --clear
```

```
# CentOS
pcs constraint remove cli-ban-myMIP-on-<NODE>
```

Listing 19.40 Rückmigration der Ressource erlauben

Wenn Sie einen Cluster mit einer größeren Zahl von Knoten haben, können Sie die Ressource anweisen, auf einen bestimmten Knoten zu migrieren. Im folgenden Beispiel ist das *Node2*:

```
# Debian/Ubuntu/openSUSE Leap
crm_resource --resource myMIP --move --node node2
```

```
# CentOS
pcs resource move myMIP node2
```

Listing 19.41 Ressource auf einen bestimmten Knoten migrieren

Unnötige Rückmigrationen verhindern

Nach einem Ausfall und der dadurch bedingten Migration der Ressourcen kann es sinnvoll sein, die Ressourcen nicht wieder zurückzumigrieren, sondern sie in ihrer neuen Heimat zu belassen. Das liegt daran, dass jeder Umzug mit einem gewissen Aufwand verbunden ist. Dieser Aufwand ist für manche Ressourcen, etwa eine IP-Adresse, vernachlässigbar gering, aber im Falle einer großen Datenbank kann die Rückmigration eine Zeit lang dauern.

Um die Rückmigration zu verhindern, passen Sie die *Resource Stickiness* an. Sie ist ein Gradmesser für den Aufwand, den eine Migration für das System bedeutet, und steht per Default auf dem Wert 0 (Null). Erhöhen Sie diesen Wert, so scheut Pacemaker vor der Migration zurück, solange kein zwingender Grund dafür vorliegt.

Ein zwingender Grund ist zum Beispiel der Ausfall eines Knotens oder ein direktes Kommando des Administrators. So ändern Sie den Default-Wert für die *Resource Stickiness*:

```
# Debian/Ubuntu/openSUSE Leap
crm configure rsc_defaults resource-stickiness=100

# CentOS
pcs resource defaults update resource-stickiness=100
```

Listing 19.42 Die »Resource Stickiness« erhöhen

19.6.2 Hochverfügbarkeit am Beispiel von Apache

Ihre erste Ressource, die IP-Adresse 10.0.0.10, ist jetzt konfiguriert, aber das ist natürlich noch nicht abendfüllend. Als Nächstes erweitern Sie den Cluster um einen Apache-Webserver, der auf Ihre Cluster-IP-Adresse »lauscht«. Installieren Sie, falls es bisher noch nicht geschehen ist, den Apache-Webserver, und ändern Sie in der Konfigurationsdatei */etc/apache2/ports.conf* unter Debian und Ubuntu, in */etc/apache2/listen.conf* unter openSUSE Leap und in */etc/httpd/conf/httpd.conf* unter Cent-OS, die Zeile `Listen 80`. Diese sollte anschließend so aussehen wie in Listing 19.43:

```
#vorher:
Listen 80

#nachher:
Listen 127.0.0.1:80
Listen 10.0.0.10:80
```

Listing 19.43 Apache auf die Cluster-IP- und »localhost«-Adresse konfigurieren

Wenn Sie Apache jetzt testweise auf beiden Knoten manuell (neu) starten, wird er auf einem Knoten funktionieren und auf dem anderen nicht – weil natürlich nur einer von beiden die Adresse 10.0.0.10 aktiviert haben kann.

Zusätzlich muss sichergestellt sein, dass das Modul *status* aktiviert ist und der Zugriff von *127.0.0.1* erlaubt ist; unter Debian und Ubuntu ist dies der Standard. Unter openSUSE Leap müssen Sie den Befehl `a2enmod status` ausführen.

Stoppen Sie auf beiden Knoten den Apache wieder (`systemctl stop apache2`), und entfernen Sie ihn aus dem automatischen Start (Runlevel) mittels `systemctl disable apache2`, denn das Starten und Stoppen des Dienstes soll von nun an Pacemaker übernehmen.

Debian/Ubuntu

Konfigurieren Sie mit dem folgenden Befehl auf einem Ihrer Clusterknoten die Apache-Ressource. Der Befehl steht eigentlich in einer einzigen Zeile, wurde aber hier aus Platzgründen mit »\« umbrochen:

```
daniel@node1:~$ sudo crm configure primitive Webserver \
ocf:heartbeat:apache \
params configfile=/etc/apache2/apache2.conf \
statusurl="http://localhost/server-status" \
op monitor interval=1m
```

Listing 19.44 Webserver-Ressource zur Clusterkonfiguration hinzufügen

Wenn Sie sich nun die Ausgabe von `crm status` anzeigen lassen, gibt es zwei Möglichkeiten. Hier sehen Sie die erste Möglichkeit:

```
[...]
myMIP (ocf::heartbeat:IPaddr2):      Started node1
Webserver (ocf::heartbeat:apache):   Started node1
```

Listing 19.45 Cluster mit zwei Ressourcen

Der Webserver wurde, wie die IP-Adresse, auf Node1 gestartet und konnte deshalb seine Arbeit aufnehmen.

Die zweite Möglichkeit ist, dass Sie eine Fehlermeldung sehen. In diesem Fall hat Pacemaker versucht, den Apache auf dem Knoten zu starten, auf dem die IP-Adresse *myMIP* gerade nicht lief. Pacemaker weiß noch nicht, dass der Apache nur in Kombination mit der IP-Adresse auf dem gleichen Knoten sinnvoll einsetzbar ist.

CentOS

Bevor Sie die Clusterressource anlegen, müssen Sie noch eine Besonderheit von CentOS ausgleichen. Die Ressource prüft nämlich den Status des Apache-Webservers und setzt dafür das Apache-Modul *mod_status* voraus. Dieses wird zwar von CentOS standardmäßig geladen, allerdings ohne Konfiguration. Um dies zu ändern, erzeugen Sie die Datei `/etc/httpd/conf.d/status.conf` mit dem Inhalt aus Listing 19.46:

```
<Location "/server-status">
    SetHandler server-status
    Require host localhost
</Location>
```

Listing 19.46 Inhalt der Datei »/etc/httpd/conf.d/status.conf«

Nun kann die Clusterressource erzeugt werden. Führen Sie dafür den nachstehenden Befehl in einer Zeile aus:

```
daniel@node1:~$ sudo pcs resource create Webserver \  
ocf:heartbeat:apache \  
configfile=/etc/httpd/conf/httpd.conf \  
statusurl="http://localhost/server-status" \  
op monitor interval=1m
```

Listing 19.47 Webserver-Ressource zur Clusterkonfiguration hinzufügen

Bitte beachten Sie, dass Sie bei der Übersetzung von Befehlen von `crmsh` nach `pcs` nicht nur die Pfadkorrektur vornehmen, sondern auch die Direktive `params` entfernen müssen.

Neben der Pfadkorrektur darf beim `pcs` nicht das Schlagwort `params` verwendet werden. Beachten Sie dies, falls Sie die Konfiguration von `crmsh` nach `pcs` übersetzen müssen.

openSUSE Leap

Leider gibt es einen Bug in der von openSUSE Leap ausgelieferten OCF-Ressource. Wenn zusätzliche Parameter übergeben werden, verhindert dieser Bug, dass sie korrekt genutzt werden. Da standardmäßig aber bereits der Pfad zur Konfigurationsdatei und auch die Prüfung des Status enthalten sind, ist dies kein Problem. Führen Sie einfach den folgenden Befehl aus, um die Clusterressource zu erstellen:

```
daniel@leap:~> sudo crm configure primitive Webserver \  
ocf:heartbeat:apache \  
op monitor interval=1m
```

Listing 19.48 Webserver-Ressource zur Clusterkonfiguration hinzufügen

Constraints: Abhängigkeiten im Cluster

Pacemaker muss lernen, dass die Ressource *Webserver* von der Ressource *myMIP* abhängig ist. *Webserver* kann nur gestartet werden, wenn *myMIP* auf dem gleichen Knoten läuft. Und nicht nur das: Die IP-Adresse muss auch bereits gestartet sein, bevor Apache an der Reihe ist. Es gibt also zwei Abhängigkeiten:

1. Die Ressourcen dürfen nicht auf unterschiedlichen Knoten gestartet werden. Wird eine Ressource migriert, muss automatisch auch die andere umziehen.
2. Eine Ressource (*myMIP*) muss immer vor der anderen (*Webserver*) gestartet werden.

Damit Pacemaker diese Abhängigkeiten beachtet, definieren Sie *Constraints*. Das sind Abhängigkeitsregeln, mit denen Sie Pacemaker die Beziehungen der einzelnen Ressourcen zueinander erklären.

Für die Regel, dass beide Ressourcen immer auf dem gleichen Clusterknoten laufen sollen, erstellen Sie einen *Colocation-Constraint*. Damit die IP-Adresse immer vor dem Webserver gestartet wird, benötigen Sie einen *Order-Constraint*. Beginnen Sie zunächst mit dem *Colocation-Constraint*:

```
# Debian/Ubuntu/openSUSE Leap
crm configure colocation WebserverMitIPAdresse INFINITY: Webserver myMIP
```

```
# CentOS
pcs constraint colocation add Webserver with myMIP INFINITY
```

Listing 19.49 Colocation-Constraint

Der Constraint erhält einen eigenen, möglichst sprechenden Namen, im Beispiel lautet er `WebserverMitIPAdresse`. Danach konfigurieren Sie, wie stringent Pacemaker diese Regel handhaben soll. Das Schlüsselwort `INFINITY` bedeutet dabei, dass keine Ausnahme zulässig ist. Hinter dem Doppelpunkt stehen die Namen der Ressourcen, für die der Constraint gelten soll, in beliebiger Reihenfolge. Jetzt kommt der Order-Constraint an die Reihe:

```
# Debian/Ubuntu/openSUSE Leap
crm configure order IPAdresseVorWebserver mandatory: myMIP Webserver
```

```
# CentOS
pcs constraint order myMIP then Webserver
```

Listing 19.50 Order-Constraint

Hier übergeben Sie unter Debian, Ubuntu und openSUSE Leap nach dem Doppelpunkt die Ressourcen in der Reihenfolge, in der Pacemaker sie starten soll. Auf CentOS mit `pcs` ist dies intuitiver mit dem Schlagwort `then` gelöst. Wenn Sie nun eine der beiden Ressourcen auf einen anderen Knoten migrieren, zieht die andere Ressource ebenfalls um. Auf dem Zielsystem wird immer die IP-Adresse zuerst gestartet, danach der Webserver.

19.6.3 DRBD integrieren

Bevor es an die Clusterkonfiguration ging, hatten Sie das gespiegelte Filesystem bereits so weit konfiguriert, dass Sie die DRBD-Ressource `mydrbd` jetzt nur noch in den Cluster einbinden müssen.

In der folgenden Schritt-für-Schritt-Anleitung erstellen Sie die benötigten Ressourcen und Constraints. Geben Sie zunächst die beiden folgenden Befehle ein (jeweils auf einer Zeile, die Befehle sind hier im Buch aufgrund der Seitenbreite umbrochen):

```
# Debian/Ubuntu/openSUSE Leap
crm configure primitive WebDRBD ocf:linbit:drbd params drbd_resource=mydrbd \
op monitor interval=1m role=Master op monitor interval=59s role=Slave
```

```
crm configure ms WebDaten WebDRBD meta master-max=1 master-node-max=1 \
clone-max=2 clone-node-max=1 notify=true
```

```
# CentOS
pcs resource create WebDRBD ocf:linbit:drbd drbd_resource=mydrbd promotable \
promoted-max=1 promoted-node-max=1 clone-max=2 clone-node-max=1 notify=true
```

Listing 19.51 DRBD-Ressource einbinden und Master-Slave-Verhältnis konfigurieren

Der erste der beiden Befehle integriert unter dem Namen `WebDRBD` grundlegend das gespiegelte Dateisystem `myDRBD` in den Cluster. Der zweite erzeugt eine *Master/Slave-Ressource* namens `WebDaten`. Pacemaker weiß nun, dass die DRBD-Ressource nur auf einem der beiden Knoten den Status *primary* haben darf. Die Ausgabe von `crm status` (CentOS: `pcs status`) sieht nun gekürzt so aus:

```
Full List of Resources:
* myMIP (ocf:heartbeat:IPaddr2): Started node1
* Webserver (ocf:heartbeat:apache): Started node1
* Clone Set: WebDaten [WebDRBD] (promotable):
  * Promoted: [ node1 ]
  * Unpromoted: [ node2 ]
```

Listing 19.52 Die Master/Slave-Ressourcen wurden eingebunden.

Die Ressource `WebDRBD` wird in dieser Übersicht nicht eigens angezeigt, da sie integraler Bestandteil der Master/Slave-Ressource `WebDaten` ist.



Neustart?

Es kann vorkommen, dass sich das Setup verschluckt. Ein Neustart von Corosync und Pacemaker hilft dabei, die Konfusionen zu entwirren (manchmal darf es auch ein Reboot sein). Verzagen Sie also nicht zu lange, sondern versuchen Sie, die Nodes mit einem kräftigen Tritt vors Schienbein davon zu überzeugen, das zu tun, was Sie wollen – im Zweifelsfall auch mit einem Reboot der Nodes.

Pacemaker kennt jetzt das Dateisystem, aber es ist noch nicht gemountet. Dafür ist eine weitere Ressource zuständig – in diesem Beispiel heißt sie `WWWMount`. So legen Sie sie an (geben Sie den folgenden Befehl auf einer einzigen Zeile ein):

```
# Debian/Ubuntu/openSUSE Leap
crm configure primitive WWWMount ocf:heartbeat:Filesystem params \
device="/dev/drbd0" directory="/srv/www-data" fstype="ext4"
```

```
# CentOS
pcs resource create WWWMount ocf:heartbeat:Filesystem \
device="/dev/drbd0" directory="/srv/www-data" fstype="ext4"
```

Listing 19.53 Filesystem-Ressource zum Mounten des DRBD-Dateisystems anlegen

Damit haben Sie festgelegt, dass das Device `/dev/drbd0`, das mit dem Dateisystem Ext4 formatiert ist, im Mountpunkt `/srv/www-data` eingehängt werden soll. Wenn Pacemaker diese Ressource auf dem Knoten startet, auf dem DRBD gerade den Status *secondary* hat, erhalten Sie nun eine Fehlermeldung.

Ignorieren Sie die Meldung für den Moment, denn das Problem wird nach der Konfiguration der Constraints verschwinden. Startet die Mountpoint-Ressource zufällig auf dem Knoten, auf dem DRBD *primary* ist, wird das Dateisystem gemountet, wie Sie mit dem Befehl `df -h` schnell herausfinden können:

```
root@node1:~# df -h
Dateisystem                Größe Benutzt Verf. Verw% Eingehängt auf
[...]
/dev/drbd0                  7,8G   18M  7,4G   1% /srv/www-data
```

Listing 19.54 Das DRBD-Dateisystem ist gemountet.

Auch mit `crm status` können Sie die Mountpoint-Ressource *WWWMount* nun sehen:

```
[...]
Full List of Resources:
* myMIP (ocf:heartbeat:IPaddr2): Started node1
* Webserver (ocf:heartbeat:apache): Started node1
* Clone Set: WebDaten [WebDRBD] (promotable):
  * Promoted: [ node1 ]
  * Unpromoted: [ node2 ]
* WWWMount (ocf:heartbeat:Filesystem): Started node1
[...]
```

Listing 19.55 Auch in der CRM-Statusübersicht wird nun das DRBD-Dateisystem angezeigt.

Sie haben jetzt alle Ressourcen definiert, die Sie benötigen. Der letzte Schritt besteht darin, Pacemaker über die Abhängigkeiten zu informieren. Dabei gibt es zwei Colocation-Constraints und zwei Order-Constraints:

1. Das Dateisystem darf nur auf dem Knoten gemountet werden, auf dem das DRBD-Device als Master, das heißt im *primary*-Modus, läuft.

Das Kommando für diesen Colocation-Constraint lautet:

```
# Debian/Ubuntu/openSUSE Leap
crm configure colocation WebMitDRBD INFINITY: WWWMount WebDaten:Master

# CentOS
pcs constraint colocation add WWWMount with WebDRBD-clone INFINITY \
with-rsc-role=Master
```

Listing 19.56 Colocation-Constraint: Dateisystem und DRBD-Master auf dem gleichen Knoten

2. Auch beim Mounten ist die Reihenfolge wichtig: Erst wenn das DRBD-Device in den *primary*-Modus geschaltet wurde, darf das Filesystem eingebunden werden. Diesen Order-Constraint legen Sie mit dem folgenden Kommando fest:

```
# Debian/Ubuntu/openSUSE Leap
crm configure order MountenNachDRBDPrimary INFINITY: WebDaten:promote \
WWWMount:start
```

```
# CentOS
pcs constraint order promote WebDRBD-clone then start WWWMount
```

Listing 19.57 Order-Constraint: erst DRBD auf Master schalten, dann mounten

3. Das DRBD-Dateisystem muss auf dem Knoten gemountet werden, auf dem der Apache-Webserver läuft. Der Befehl für diesen Colocation-Constraint lautet:

```
# Debian/Ubuntu/openSUSE Leap
crm configure colocation WebServerUndDateisystem INFINITY: Webserver WWWMount
```

```
# CentOS
pcs constraint colocation add Webserver with WWWMount INFINITY
```

Listing 19.58 Colocation-Constraint: Mount auf gleichem Knoten wie Webserver

4. Erst wenn das Filesystem gemountet ist, darf der Apache-Webserver gestartet werden:

```
# Debian/Ubuntu/openSUSE Leap
crm configure order WebServerNachDateisystem INFINITY: WWWMount Webserver
```

```
# CentOS
pcs constraint order WWWMount then Webserver
```

Listing 19.59 Order-Constraint: erst mounten, dann Webserver starten

Spätestens jetzt laufen alle Ressourcen auf dem gleichen Knoten:

```
root@node1:~# crm status      ### CentOS: pcs status
[...]
Full List of Resources:
* myMIP (ocf:heartbeat:IPaddr2): Started node1
* Webserver (ocf:heartbeat:apache): Started node1
* Clone Set: WebDaten [WebDRBD] (promotable):
  * Promoted: [ node1 ]
  * Unpromoted: [ node2 ]
* WWWMount (ocf:heartbeat:Filesystem): Started node1
[...]
```

Listing 19.60 Alle Ressourcen sind auf dem gleichen Knoten gestartet.

Wenn Sie jetzt eine beliebige Ressource zwangsmigrieren, schwenken die IP-Adresse, das DRBD-Filesystem und der Apache – und zwar in genau dieser Reihenfolge, denn dafür sorgen die Order-Constraints – auf den anderen Clusterknoten um:

```
# Debian/Ubuntu/openSUSE Leap
root@node1:~# crm_resource --resource Webserver --move --node node2; crm status
```

```
# CentOS
[root@node1 ~]# pcs resource move Webserver node2 ; pcs status
```

```
# Ausgabe:
[...]
Online: [ node1 node2 ]
```

Full list of resources:

```
myMIP (ocf::heartbeat:IPaddr2): Started node2
myMIP2 (ocf::heartbeat:IPaddr2): Started node2
Webserver (ocf::heartbeat:apache): Started node2
Master/Slave Set: WebDaten [WebDRBD]
    Masters: [ node2 ]
    Slaves: [ node1 ]
WWWMount (ocf::heartbeat:Filesystem): Started node2
```

Listing 19.61 Die Migration einer Ressource führt zur Migration aller abhängigen Ressourcen.

Jetzt ist es an der Zeit, auch dem Webserver mitzuteilen, dass er doch bitte die Webseiten von `/srv/www-data` ausliefern soll.

Damit der Apache die neue Konfiguration auch verwendet, müssen wir, da wir ja auf einem Cluster arbeiten, entweder die Ressource verschieben oder folgenden Befehl absetzen, damit die Ressource (also der Apache) neu gestartet wird:

```
# Debian/Ubuntu/openSUSE Leap
crm resource restart Webserver
# CentOS
pcs resource restart Webserver
```

Listing 19.62 Neuladen der Webserver-Konfiguration im Cluster

19.6.4 Fencing

Vielleicht ist Ihnen im Node-Menü der CRM-Shell der Punkt *fence* aufgefallen. *Fencing* ist ein Prozess, einen Knoten im Fehlerfall sicher aus dem Cluster herauszutrennen. Stellen Sie sich folgende Situation vor:

1. Sie haben einen Datenbank-Cluster aus drei Knoten, die auf dem gleichen Datenbestand arbeiten.
2. Einer der Knoten antwortet nicht mehr. Der Cluster schwenkt auf die verbliebenen beiden Knoten um und arbeitet weiter.
3. Der ausgefallene Knoten hatte nur einen »Hänger«, arbeitet nach einigen Minuten aber weiter. Da er nicht mehr zum Cluster gehört, aber noch Zugriff auf den Datenbestand hat (Shared Filesystem!), wird er mit hoher Wahrscheinlichkeit beim nächsten schreibenden Zugriff die Datenbank in die Inkonsistenz und den Admin in den Wahnsinn treiben.

Einen solchen Fall nennt man eine *Split-Brain*-Situation. Um sie zu entschärfen, gibt es die Fencing-Prozesse. Fencing soll die Beschädigung von Datenbeständen verhindern. Wenn Sie keinen Datenbestand in Ihrem Cluster haben, der in einer Split-Brain-Situation Schaden nehmen könnte, können Sie auf ein Fencing verzichten, aber solche Szenarien sind selten. Stellen Sie sich zur Kontrolle einfach die Frage: »Was kann ein Amok laufender Clusterknoten im schlimmsten Fall anrichten?«

Die bekannteste Fencing-Maßnahme ist STONITH (»Shoot The Other Node In The Head«). Dabei versuchen die beiden Teile des Split Brains jeweils, die andere Hälfte vollständig zu deaktivieren. Diese Deaktivierung wird fast immer auf der Hardwareebene gelöst, etwa mit fernschaltbaren Stromanschlüssen. Sie trennen auf ein Softwarekommando hin die Spannungsversorgung und sorgen so für einen zweifelsfrei definierten Zustand des abtrünnigen Clusterknotens. Da diese STONITH-Devices auf der Hardwareseite sehr variantenreich und herstellerspezifisch sind, können wir hier keine allgemeingültigen Aussagen zum »richtigen« Fencing-Prozess machen. Denken Sie aber beim Bau Ihres Clusters daran, dass Sie sehr wahrscheinlich ein funktionierendes Fencing benötigen, und ziehen Sie die Dokumentation Ihrer Hardware zurate.

Kapitel 20

Virtualisierung

Serversysteme on demand installieren und betreiben zu können ist kein Wunschdenken, der Virtualisierung sei Dank. Sie erfahren in diesem Kapitel, wie Sie Desktop-Virtualisierung und Servervirtualisierung mit Live Migration in Ihrer Umgebung betreiben können.

Virtualisierung ist der wohl am stärksten wachsende Bereich der Informatik in den letzten zehn Jahren. Virtualisierung ermöglicht den parallelen Betrieb einer Vielzahl von Betriebssystemen auf nur einer physischen Hardware. Nicht nur die Konsolidierung der vorhandenen Ressourcen wird dadurch verstärkt, sondern es wird auch die spontane Erweiterung des Umfelds ermöglicht. In den Zeiten von Hochverfügbarkeit und des dauerhaften Vorhaltens von QS¹-Systemen zählt die Virtualisierung von Serverfarmen mehr und mehr zum Alltag eines Systemadministrators.

20.1 Einleitung

Bei der Virtualisierung unterscheidet man Gast- und Wirt-Systeme. Die Wirt-Systeme sind die physische Hardware. Gast-Systeme (auch Gäste oder VMs² genannt) laufen auf dem Wirt und nutzen dessen physische Hardware. Das Aufteilen der physischen Ressourcen geschieht mithilfe eines *Hypervisors*, der die Ressourcen intelligent verteilt. Jedem Gast wird dabei ein vollständiges isoliertes System zur Verfügung gestellt.

Drei unterschiedliche Betriebsarten werden bei der Virtualisierung unterschieden:

► **Emulation**

Hierbei wird die Gast-Hardware vollständig auf dem Wirt simuliert. Damit kann jede Rechnerarchitektur zur Verfügung gestellt werden, auch wenn der Wirt diese gar nicht besitzt. Diese Form der Virtualisierung ist äußerst rechenintensiv und somit auch nicht sehr performant.

► **Virtualisierung (oder Vollvirtualisierung)**

Die Hardware des Wirt-Systems wird den Gast-Systemen teilweise als virtuelle Hardware

1 QS = Qualitätssicherung

2 VM = virtuelle Maschine

zur Verfügung gestellt. Dazu werden spezielle Prozessoren benötigt, die das Durchreichen in die virtuelle Umgebung unterstützen – zum Beispiel *AMD-V (Pacifica)* oder *Intel VT (Vanderpool)*. Gast-Systeme können unangepasst betrieben werden, allerdings nur mit der gleichen CPU-Architektur des Wirt-Systems. Vollvirtualisierte Maschinen sind deutlich performanter als emulierte virtuelle Maschinen.

► **Paravirtualisierung**

Die Paravirtualisierung arbeitet hardwarenah. Bei dieser Form wird keine Hardware virtualisiert oder emuliert. Das Gast-System kommuniziert mit dem Wirt-System über eine abstrahierte Verwaltungsschicht. So ist das Höchstmaß an Performance möglich, allerdings müssen die Gast-Betriebssysteme speziell angepasst werden, damit sie mit der abstrahierten Verwaltungsschicht zusammenarbeiten können.

Es gibt eine Vielzahl an Virtualisierungslösungen. Zu den gängigen Systemen gehören:

- Xen
- KVM
- OpenVZ
- Parallels
- VMware
- Virtuozzo
- VirtualBox
- Citrix
- Microsoft Virtual PC

Die Anwendungsgebiete der Virtualisierung umfassen dabei nicht nur die Serverinfrastruktur. Auch die lokale Desktop-Virtualisierung spielt im Alltag mittlerweile eine große Rolle. In diesem Kapitel erfahren Sie alles zum Thema Virtualisierung, und Sie lernen, wie Sie diese komfortabel auf Ihrem Desktop-System einsetzen können und wie Sie Ihre Serverinfrastruktur auf virtuellen Maschinen umstellen, um so Ihre zur Verfügung stehenden Ressourcen noch effizienter einsetzen zu können.

20.2 Für den Sysadmin

Eine häufige Anforderung an Sysadmins ist das berühmte »mal eben«. Dies kann sich auf den Test einer neuen Softwareversion beziehen oder auf die Prüfung, ob eine Konfiguration auch auf neueren Betriebssystemen lauffähig ist. Nicht jede Anforderung rechtfertigt die Anschaffung eines eigenen QS-Systems, da der Sysadmin von heute sich auch der Kostenfrage stellen muss. Ressourcen müssen möglichst effizient eingesetzt werden, damit die Gesamtkalkulation nicht aus den Fugen gerät.

Für solche »mal eben«-Anwendungszwecke bietet sich der Einsatz einer lokalen VM an. Eines der bekannteren Programme hierfür ist *VirtualBox*. Das von *Oracle* entwickelte Programm steht (in der aktuellen Version 5) in fast allen Distributionen zur Verfügung. Es wird sowohl in einer kommerziellen Variante als auch in einer Open-Source-Variante angeboten. Diese Varianten wurden bis zur Version 4 in zwei unterschiedlichen Paketen angeboten, als *OSE (Open Source Edition)* und als *non-free (nicht freie)* Version. Der nicht freie Zweig wurde mittlerweile als Erweiterungspaket ausgelagert, sodass nur noch eine Version existiert, die um den kommerziellen Teil erweitert werden kann.

Wenn Ihr System die Vollvirtualisierung unterstützt, wird diese von *VirtualBox* verwendet, ansonsten werden die virtuellen Maschinen emuliert. *VirtualBox* unterstützt lediglich die x86-Architektur.

Die Open-Source-Variante enthält ein paar Einschränkungen, die Sie gegebenenfalls berücksichtigen müssen:

► **Grafik**

Die Grafik der VM kann nur in 2D beschleunigt werden – was dem Sysadmin aber genügen sollte.

► **USB**

Das Durchreichen von USB-Geräten in das Gast-System ist nicht möglich.

► **PXE**

Das Starten von Betriebssystemen über das Netzwerk für Intel-Karten ist nicht möglich.

Installieren Sie das Paket *virtualbox* aus den Paketquellen. Die Installation einer VM muss nicht zwingend wie bei einem physischen System über das CD-ROM-Laufwerk erfolgen. Sie können auch einfach ein ISO-Image verwenden, das bereits auf Ihrer Festplatte liegt.

Dafür müssen Sie dieses Image aber vorab den *Medien* von *VirtualBox* bekannt machen. Starten Sie *VirtualBox*, und fügen Sie ein Installationsmedium über das Menü DATEI • MANAGER FÜR VIRTUELLE MEDIEN hinzu, wie in Abbildung 20.1 dargestellt.

Im Startbildschirm von *VirtualBox* können Sie über den Button NEU eine neue virtuelle Maschine anlegen. Durchlaufen Sie den selbsterklärenden Dialog, und passen Sie die Konfiguration entsprechend Ihrer Umgebung an. Wählen Sie bei der Auswahl des CD-ROM-Laufwerks das vorher angelegte Installationsimage aus.

Sobald Sie Ihre neu angelegte VM starten, bootet diese von dem angegebenen Image. Die Installation findet nun zwar in einem Fenster, aber ansonsten wie gewohnt statt.

Maus/Tastatur aus der VM lösen

Um die Eingabegeräte aus der VM zu lösen, betätigen Sie die rechte `[Strg]`-Taste. Diese Taste kann über das Menü DATEI • GLOBALE EINSTELLUNGEN im Reiter EINGABE angepasst werden.



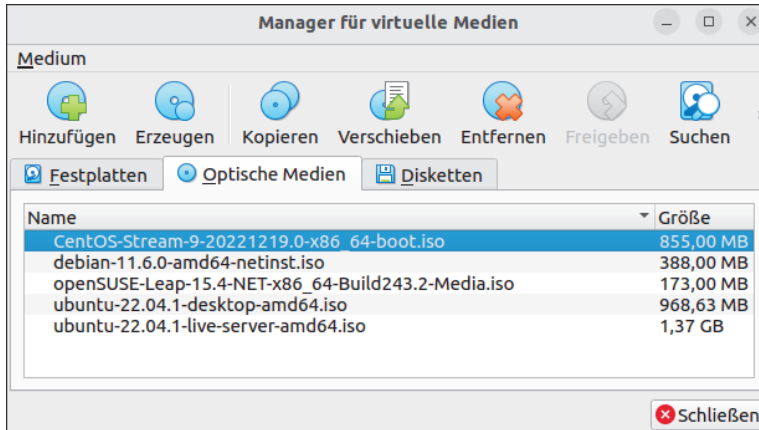


Abbildung 20.1 Medien anlegen

Selbstverständlich können Sie die gewählten Parameter der VM auch nachträglich ändern (bis auf die bereits angesprochene Festplattengröße). Dafür muss die VM aber ausgeschaltet sein. Der Startbildschirm von VirtualBox bietet dafür eine direkte Auswahl, wie Sie in Abbildung 20.2 sehen. Über einen Klick auf die Überschriften gelangen Sie in den Konfigurationsmodus.

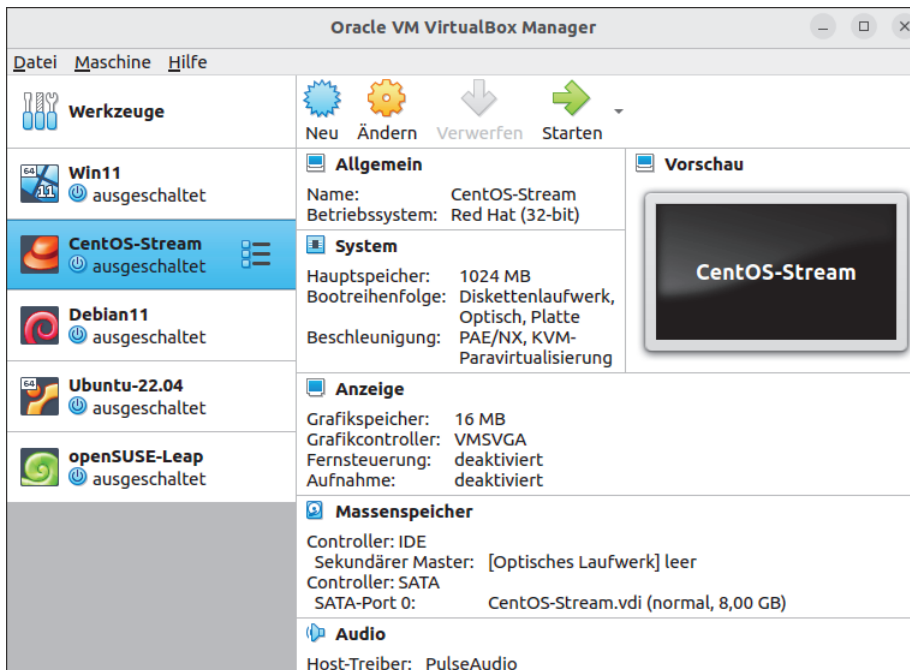


Abbildung 20.2 Übersicht über die virtuellen Maschinen

Neben dem Betrieb eines fremden Betriebssystems auf Ihrem lokalen System bietet Virtual-Box den Vorteil, dass Sie die VM auch in Ihr lokales Netz einbinden können. VirtualBox stellt dabei eine vielfältige Netzwerkkonfiguration zur Verfügung (siehe Abbildung 20.3):

- ▶ **NAT**
Zwischen Wirt und Gast wird ein lokales Netz aufgebaut, Anfragen vom Gast ins Netz werden über die IP-Adresse des Wirts umgesetzt. Vom Wirt, aus dessen Netzwerk oder von anderen Gästen aus kann nicht auf den Gast zugegriffen werden.
- ▶ **Netzwerkbrücke**
Das Gast-System wird über eine Brücke ins lokale Netz geschleust – so, als ob es ein physisches System wäre.
- ▶ **Internes Netzwerk**
Nur Gäste, denen dieses Netzwerk zugewiesen wurde, können untereinander kommunizieren – Verbindungen zum Wirt oder dessen LAN sind nicht möglich.
- ▶ **Host-only Adapter**
Eine Kommunikation zwischen Gast- und Wirt-System ist möglich – für jeden Gast wird ein eigenes virtuelles Interface angelegt (zum Beispiel *vboxnet0*).
- ▶ **Generischer Treiber**
Dies ist ein experimenteller Modus, der es erlaubt, Gäste von unterschiedlichen Wirten untereinander kommunizieren zu lassen.
- ▶ **NAT-Netzwerk**
Wie NAT, nur dass Gäste untereinander kommunizieren können.
- ▶ **nicht angeschlossen**
Es gibt keine Netzwerkverbindung.

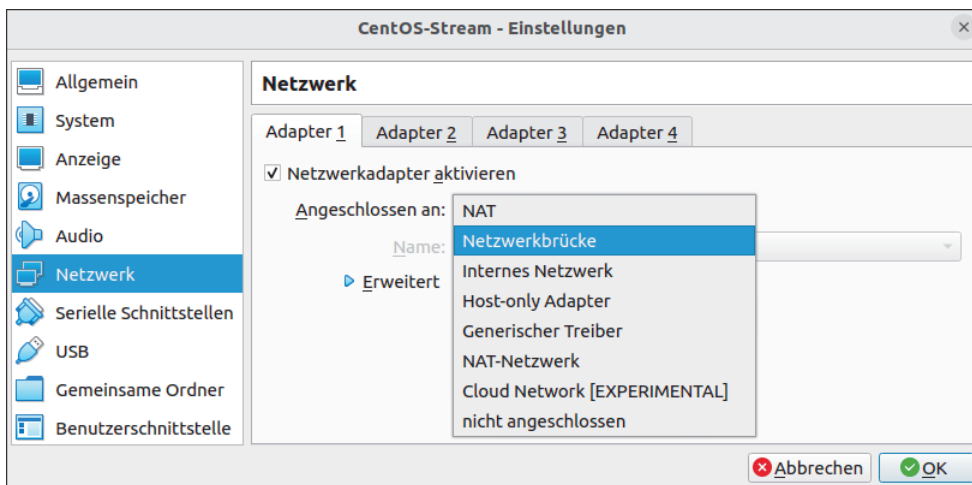


Abbildung 20.3 Netzwerkkonfiguration

Selbstverständlich können Sie auch (je nach zur Verfügung stehender Hardware) mehrere virtuelle Maschinen gleichzeitig betreiben. Durch die vielfältigen Netzwerkooptionen können Sie so leicht eine eigene DMZ für Ihre virtuellen Maschinen aufbauen oder diese über Ihr System direkt mit dem lokalen Netzwerk verbinden.



Bei der Erstellung einer VM können Sie wählen, ob die Größe der Festplatte direkt alloziert wird oder ob diese dynamisch wachsen soll. Allozierter Speicher hat Geschwindigkeitsvorteile und bewahrt Sie vor dem plötzlichen Überlauf Ihrer physischen Festplatte. Dynamisch wachsende Festplatten hingegen belegen nur den realen Speicherplatz, den die VM wirklich verbraucht. Beachten Sie dies bei der Erstellung Ihrer virtuellen Maschinen.

Ein weiterer großer Vorteil von VirtualBox ist die Exportfunktionalität. Damit haben Sie die Möglichkeit, einmal eingerichtete virtuelle Maschinen zu exportieren, sie auf einem anderen System mit VirtualBox zu importieren und direkt ohne Neuinstallation oder Neukonfiguration zu betreiben. Sie exportieren eine VM über das Menü DATEI • APPLIANCE EXPORTIEREN. Wählen Sie anschließend die zu exportierende VM und den Speicherort des Exports aus. Nach dem Export finden Sie im angegebenen Pfad zwei Dateien. Bei der Datei mit der Endung *.ovf* handelt es sich um ein XML-Dokument, das die Konfiguration der VM beschreibt. Die Datei mit der Endung *.vmdk* ist die virtuelle Festplatte der VM. Kopieren Sie diese Dateien auf das neue System, und importieren Sie sie in VirtualBox über das Menü DATEI • APPLIANCE IMPORTIEREN. VirtualBox passt die VM der neuen Umgebung an. Anschließend können Sie die VM direkt starten, als wäre sie auf dem System installiert worden.

20.3 Servervirtualisierung

Neben dem Bereich der Virtualisierung in Desktop-Umgebungen ist die Servervirtualisierung das Kernaufgabengebiet des Sysadmins. Die Servervirtualisierung findet hauptsächlich im Bereich von Rechenzentren und *Cloud Computing* statt.

Die Virtualisierungssoftware selbst stellt dabei nur eine Schnittstelle zum Betrieb bereit. Die Verwaltung der virtuellen Maschinen erfolgt über separate Software. Der große Funktionsumfang der Virtualisierungslösungen wird durch die Vielzahl der Parameter deutlich, die die jeweiligen Lösungen verarbeiten können. Für einzelne Systeme können Sie das Anlegen, Starten, Stoppen und Verwalten der Systeme ohne Probleme mit den Lösungen selbst regeln. In großen Umgebungen ist eine Managementsoftware aber unverzichtbar.

Zurzeit ist die Virtualisierungswelt im Umbruch. Die einstigen Verfechter von Xen wandern nach und nach in Richtung KVM ab. Es bleibt abzuwarten, wie weit sich dieser Trend fortsetzt oder ob Xen vielleicht doch noch eine Renaissance erleben wird.

In diesem Abschnitt erfahren Sie alles zur Virtualisierungslösung KVM, und Sie erhalten einen Einstieg in Xen. Die vorgestellte Managementsoftware *libvirt* mit *virt-install*, *virsh* und Co. kann sowohl mit KVM als auch mit Xen betrieben werden.

20.3.1 KVM

Die *Kernel-based Virtual Machine* (KVM) läuft, wie der Name bereits angibt, im Kernel und stellt eine Hardwareschnittstelle zur Virtualisierung bereit. Diese Virtualisierungslösung basiert auf x86-Hardware. KVM benötigt für den Betrieb spezielle Prozessoren, die mit einem speziellen Hardware-Hypervisor ausgestattet sind. Diese Prozessoren des Typs *AMD-V (Pacifica)* oder *Intel VT (Vanderpool)* sind für KVM zwingend erforderlich. Durch die Unterstützung des Hardware-Hypervisors verfügt KVM über eine gute Performance und ermöglicht den nativen (unangepassten) Betrieb von virtuellen Maschinen. Das im Jahre 2008 von Red Hat gekaufte israelische Unternehmen *Qumranet* entwickelte KVM. Seit 2007 ist KVM Bestandteil des Linux-Kernels und kann somit auf fast allen Distributionen betrieben werden. Durch den Betrieb im Kernel stellt sich die Virtualisierung mit KVM wie in Abbildung 20.4 gezeigt dar.

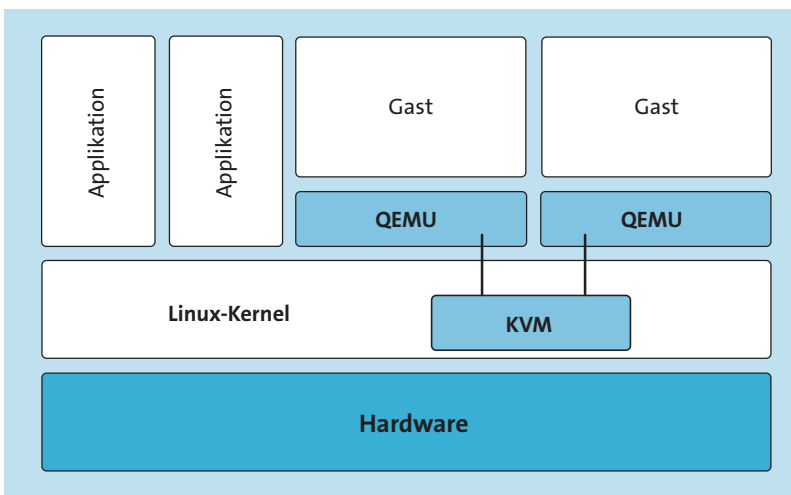


Abbildung 20.4 Betriebskonzept von KVM

Überprüfen Sie vorab, ob Ihr System einen Hardware-Hypervisor besitzt. Lesen Sie dafür die Datei `/proc/cpuinfo` aus, und prüfen Sie, ob die Statusindikatoren (*Flags*) `vmx` oder `svm` vorhanden sind. Dies können Sie mit einem einfachen regulären Ausdruck erreichen. Erhalten Sie bei dem in Listing 20.1 aufgeführten Befehl keine Ausgabe, ist der Prozessor nicht in der Lage, mit KVM zu virtualisieren.

```
root@example:/# grep -E '^flags.*\b(vmx|svm)\b' /proc/cpuinfo
```

Listing 20.1 Prozessoranforderung prüfen

Wenn Ihr Test bestätigt, dass Ihr System KVM nutzen kann, installieren Sie als Nächstes das Metapaket `qemu-kvm` aus den Paketquellen. Die benötigten abhängigen Pakete werden direkt mit installiert. Zusätzlich sollten Sie unter Debian/Ubuntu die Pakete `libvirt-clients`

und *libvirt-daemon-system* (unter CentOS/openSUSE nur *libvirt*), *virtinst* (unter CentOS/openSUSE *virt-install*) und *bridge-utils* installieren. In CentOS ist das Paket *bridge-utils* nicht mehr vorhanden. Sie können es aber über die abwärtskompatible Version von CentOS 7 installieren:

```
$ dnf install http://mirror.centos.org/centos/7/os/x86_64/Packages/bridge-
utils-1.5-9.el7.x86_64.rpm
```

Bei entsprechenden Prozessoren werden die benötigten Module meist automatisch geladen. Dies sollten Sie ebenfalls vorab wie folgt prüfen:

```
root@example:/# lsmod | grep kvm
kvm_intel          212992  0
kvm                593920  1 kvm_intel
irqbypass         16384   1 kvm
```

Listing 20.2 »KVM«-Module: Intel-CPU

Bei AMD-Prozessoren würde entsprechend das Modul *kvm_amd* geladen sein.



Hardwarevirtualisierung im BIOS aktivieren

Die Hardwarevirtualisierung kann im BIOS deaktiviert sein. In diesem Zustand startet der KVM-Dienst nicht. Die Ausgabe von *dmesg* enthält dann die Meldung *kvm: disabled by bios*.

Eine Übersicht darüber, welche Gast-Systeme KVM unterstützt, finden Sie auf der KVM-Projektseite unter der URL www.linux-kvm.org/page/Guest_Support_Status.

Wie bereits erläutert wurde, stellt KVM nur die Linux-Kernelschnittstelle zur Verfügung. Die Virtualisierungsumgebung wird durch QEMU realisiert. Bei QEMU handelt es sich um eine Emulationssoftware, deren Quellcode auch zur Grundlage der Emulierung von virtuellen Maschinen in VirtualBox dient.

»virtio«

Aufgrund der Vollvirtualisierung von KVM sind gerade die *I/O*³-Verarbeitungen kritisch, da diese äußerst rechenintensiv sind. Seit der KVM-Version 60 und dem Linux-Kernel 2.6.25 gibt es eine weitere Möglichkeit, *I/O*-Zugriffe durch KVM verarbeiten zu lassen. Das von Rusty Russell entwickelte Tool *virtio* stellt eine Möglichkeit bereit, *I/O*-Zugriffe paravirtualisiert zu betreiben. Durch eine weitere Abstraktionsebene werden Zugriffe einer VM paravirtualisiert auf der physischen Hardware verarbeitet. Dadurch ist ein enormer Geschwindigkeitszuwachs entstanden.

³ *Input/Output*, engl. für *Eingabe/Ausgabe*

20.3.2 Xen

Die an der britischen Universität Cambridge entstandene und heute vom Unternehmen Citrix weiterentwickelte Virtualisierungslösung Xen ist ein reiner Software-Hypervisor, auch *Virtual Machine Monitor (VMM)* genannt. Im Gegensatz zu KVM läuft Xen direkt auf der Hardware und verfolgt das Konzept der Paravirtualisierung.

Diesem Konzept liegt das Xen-Domänenkonzept zugrunde. Dabei stellt Xen selbst die *Dom0* dar. Virtuelle Maschinen werden in unprivilegierten Domänen betrieben, die man als *DomU* bezeichnet.

Für die Paravirtualisierung setzt Xen, äquivalent zu KVM, auf die Hardware-Hypervisoren der Prozessortypen *AMD-V (Pacifica)* oder *Intel VT (Vanderpool)*. Sind diese nicht vorhanden, wird lediglich eine Vollvirtualisierung durch Xen geboten. Abbildung 20.5 verdeutlicht das Betriebskonzept der Xen-Virtualisierung.

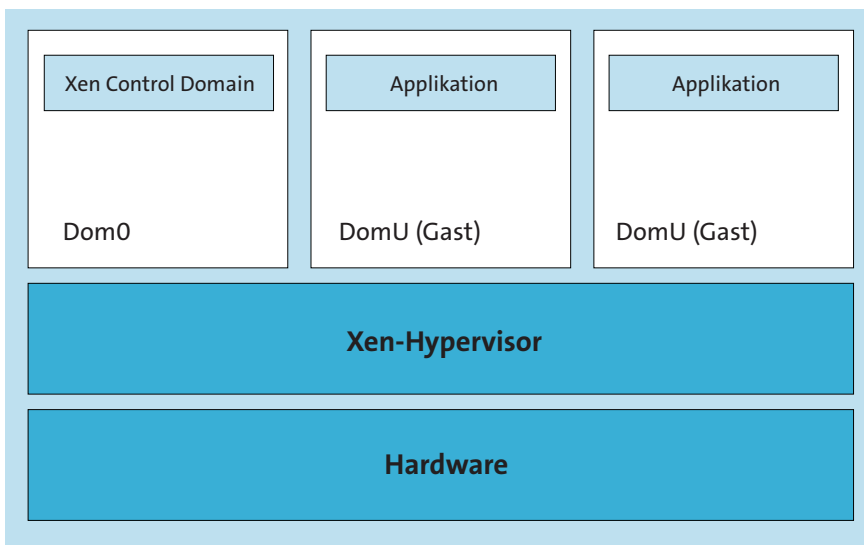


Abbildung 20.5 Betriebskonzept von Xen

Durch den Betrieb im Kernel müssen die Xen-Anpassungen für den jeweiligen Kernel vorgenommen werden. Dies ist nicht nur beschwerlich, sondern auch teilweise nicht möglich, da entsprechende Optionen nicht mehr vorhanden sind.

Dies führt dazu, dass Xen-Systeme auf veralteten Kernel-Versionen betrieben werden müssen. Darüber hinaus muss jedes Gast-System angepasst werden, damit die Vorteile der Paravirtualisierung auch genutzt werden können.

Wie schon erwähnt wurde, ist die Virtualisierungswelt im Umbruch: Einstige Unterstützer von Xen wandern in Richtung KVM ab. Es bleibt ungewiss, wie weit Xen noch entwickelt wird.

20.4 Netzwerkgrundlagen

Bevor die eigentliche Installation der virtuellen Maschinen erfolgen kann, müssen Sie einige Grundkonfigurationen vornehmen. Virtualisierungslösungen können ihren Gästen nur das anbieten, was sie auch auf dem physischen System vorfinden. Dazu zählen die Netzwerkkonfiguration und eine einheitliche Managementschnittstelle, deren Grundlagen in diesem Abschnitt näher betrachtet werden.

Bekanntlich führen viele Wege nach Rom – mindestens ebenso viele gibt es, wenn Sie sich an die Netzwerkkonfiguration einer VM machen. In den Untiefen des Internets finden sich haufenweise Anleitungen zur »richtigen« Netzwerkkonfiguration für eine Virtualisierungslösung. Zum einen werden *tun/tap*-Schnittstellen angelegt, zum anderen wird ein virtueller Switch über *vde* eingerichtet, oder es werden *Bridging*-Schnittstellen angepriesen. Neben dem Verlust des Überblicks bleibt die Frage: »Welche Konfiguration benötige ich?«

Dies hängt hauptsächlich von Ihren Anforderungen ab. Alle Virtualisierungslösungen bieten folgende grundlegenden Netzwerkkarten an:

► **Isoliertes LAN**

Hierbei wird ein eigenes lokales Netzwerksegment erstellt. Gast-Systeme, die diese Netzwerkumgebung nutzen, können untereinander kommunizieren. Ebenfalls ist die Kommunikation zum Wirt-System möglich. Netzwerkverkehr ins LAN oder darüber hinaus wird unterbunden.

► **NAT (Adressumsetzung)**

Den Gast-Systemen wird ein privater Adressbereich zugeteilt. Die Kommunikation zwischen den Gästen und zum Wirt-System ist ebenfalls möglich. Netzwerkverkehr ins LAN oder darüber hinaus wird auf die Adresse des Wirt-Systems umgesetzt. Die meisten Virtualisierungslösungen verwenden NAT als Standardeinstellung.

► **Bridging (Netzwerkbrücke)**

Die Gast-Systeme werden über die physikalische Schnittstelle des Wirt-Systems direkt mit dem LAN verbunden. Gäste im *Bridging*-Modus agieren, als wären sie mit dem realen Netzwerk verbunden.

Selbstverständlich sind diese Betriebsarten nicht statisch. Jede erdenkliche Verknüpfung ist möglich, selbst eine Erweiterung mithilfe von *VLANs*⁴.

Damit Ihre virtuellen Maschinen alle Netzwerkkarten verwenden können, müssen Sie die Netzwerkkonfiguration des Wirt-Systems anpassen. Die ersten beiden Betriebsarten sind dabei ohne weitere Konfiguration möglich. Damit Ihr System als Brückenbauer fungieren kann, wird bei der Installation das Paket *bridge-utils* automatisch mit installiert.

⁴ *Virtual Local Area Networks*, engl. für *virtuelle lokale Netzwerke*. Damit werden Netzwerke auf einem Switch so virtuell getrennt, als ob sie physikalisch mit unterschiedlichen Geräten verbunden wären.

Während der Installation wird bereits eine Brückenschnittstelle angelegt: `virbr0`. Mit dem Paket `bridge-utils` wird Ihnen auch ein Kontrollprogramm an die Hand gegeben. Um die Konfiguration zu prüfen, können Sie folgenden Befehl absetzen:

```
root@example:/# brctl show
bridge name      bridge id                STP enabled   interfaces
virbr0           8000.525400f68b43       yes           virbr0-nic
                                                         vnet0
```

Listing 20.3 Netzwerkbrücke kontrollieren

Erhalten Sie keine Ausgabe, ist die Konfiguration fehlerhaft oder unvollständig. Für die Netzwerkanbindung der virtuellen Maschinen wird ein eigener DHCP-Server bereitgestellt. Standardmäßig wird der Betrieb (analog zum Bridging) nur auf das Interface `virbr0` eingeschränkt und die DHCP-Range `192.168.122.2` bis `192.168.122.254` verwendet. Falls Ihnen dies so genügt, können Sie direkt fortfahren.

Netzwerk von A–Z

Wie Sie alles aus Ihrem Netzwerk herausholen, zeigen wir Ihnen in Kapitel 22, »Netzwerk«.



20.5 Management und Installation

Die Virtualisierungssoftware selbst stellt, wie bereits erörtert wurde, nur eine Schnittstelle zum Betrieb bereit. Die Verwaltung der virtuellen Maschinen erfolgt über separate Software. Folgende Programme sind verfügbar:

- ▶ **virsh**
Konsolen-Managementsoftware (basiert auf *libvirt*)
- ▶ **virt-manager**
GUI-Managementsoftware (basiert ebenfalls auf *libvirt*)
- ▶ **oVirt**
libvirt-basierende Web-GUI, die von Red Hat entwickelt wird. Sie wird hauptsächlich zur Data-Center-Verwaltung verwendet.
- ▶ **Ganeti**
Cluster-Virtual-Server-Managementsoftware von Google
- ▶ **openQRM**
Cloud-Computing- und Data-Center-Management-Plattform mit Xen, KVM, *VMware* und *Linux VServer* als Basis für virtualisierte Server
- ▶ **und viele weitere**

Jede Virtualisierungssoftware verfolgt ein eigenes Konzept zum Management der virtuellen Maschinen. Damit die unterschiedliche Software über eine Managementlösung administriert werden kann, wurde die bereits vorgestellte C-Bibliothek *libvirt* entwickelt.

In diesem Abschnitt stellen wir Ihnen die Möglichkeiten zur Administration von virtuellen Maschinen über KVM selbst und über die verschiedenen Managementlösungen vor, die auf *libvirt* aufsetzen.

20.5.1 Einheitlich arbeiten: »libvirt«

Bei dem von Red Hat geförderten Projekt *libvirt* handelt es sich um eine Abstraktionsschicht zur homogenen Administration unterschiedlicher Virtualisierungslösungen. Zur vereinfachten Administration und um Entwicklern solcher Lösungen eine einheitliche Schnittstelle zur Verfügung zu stellen, wurde das Projekt im Jahre 2005 gestartet.

Bei der Bibliothek *libvirt* werden Wirt-Systeme als *Nodes* und Gast-Systeme als *Domains* bezeichnet. Von *libvirt* werden die Virtualisierungslösungen *QEMU*, *KVM*, *Xen*, *VirtualBox*, *VMware ESX*, *LXC Linux Container System*, *OpenVZ*, *User Mode Linux* und *OpenNebula* unterstützt. Um dies zu ermöglichen, stellt *libvirt* eine eigene *API*⁵ bereit. Damit kann Managementsoftware, die auf *libvirt* aufsetzt, alle Virtualisierungslösungen verwalten.

Die Anbindung von *libvirt* an die eigentlichen Virtualisierungslösungen erfolgt über die Angabe einer URI. Diese spezifiziert die Virtualisierungslösung und wandelt die entsprechenden Befehle passend um. Die Managementsoftware kommuniziert über die *Public API*, und durch die Angabe der URI werden die Befehle von *libvirt* entsprechend über die *Driver API* umgesetzt. Abbildung 20.6 stellt die Arbeitsweise von *libvirt* grafisch dar.

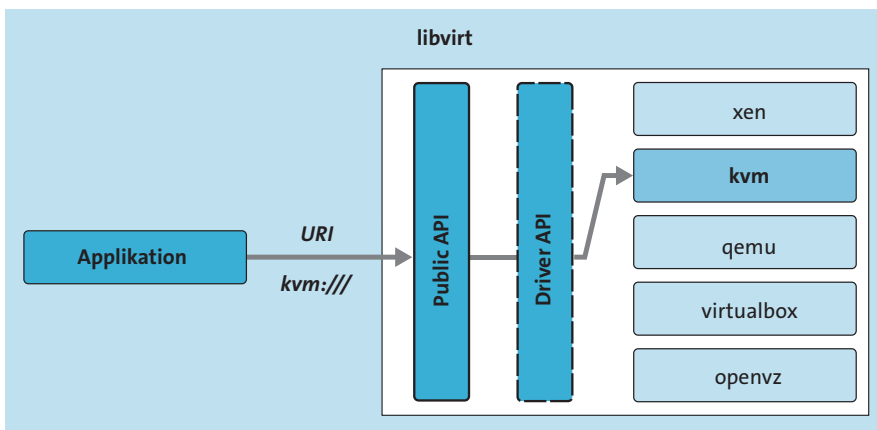


Abbildung 20.6 Arbeitsweise von »libvirt«

⁵ *Application Programming Interface*, engl. für *Programmierschnittstelle*.

Dabei kann *libvirt* die in Tabelle 20.1 dargestellten Verbindungen aufbauen.

Verbindung	Bedeutung
test	Anbindung zum Test-Hypervisor
qemu	lokale Verbindung zu <i>QEMU/KVM</i>
xen	lokale Verbindung zu Xen
vbox	lokale Verbindung zu <i>VirtualBox</i>
esx	lokale Verbindung zu <i>VMware ESX</i>
gsx	lokale Verbindung zu <i>VMware GSX</i>
<HV>+unix	lokale Verbindung über Unix-Sockets zum Hypervisor (HV)
<HV>+ssh	Verbindung zum Hypervisor (HV) über einen SSH-Tunnel
<HV>+tcp	Verbindung zum Hypervisor (HV) über einen TCP-Socket

Tabelle 20.1 »libvirt«-Verbindungstypen

Installieren Sie unter Debian/Ubuntu das Paket *libvirt-clients* und unter CentOS/SUSE *libvirt*, um die einheitliche Managementlösung *libvirt* auf Ihrer Virtualisierungsumgebung einzusetzen.

Allerdings unterstützen nicht alle Virtualisierungslösungen die Möglichkeiten von *libvirt*, bzw. *libvirt* unterstützt nicht alle Funktionen jeder Virtualisierungslösung. Der Großteil der Funktionalitäten wird aber unterstützt und stetig erweitert. Dadurch stellt *libvirt* eine homogene Umgebung zur Administration unterschiedlicher Virtualisierungslösungen zur Verfügung.

Durch die Installation und Verwaltung von virtuellen Maschinen mit *libvirt* erhält jede VM eine eigene XML-Datei. In dieser sind die Spezifikationen der VM enthalten, wie der Hypervisor oder die zugeordnete Hardware. Dies ist einer der großen Vorteile von *libvirt*. Durch die Standardisierung der Konfiguration in der XML-Datei können unterschiedliche Managementumgebungen ein und dieselbe VM administrieren, und die Änderungen sind in allen Umgebungen vorhanden.

Konfiguration im Detail

Die Konfigurationen der virtuellen Systeme werden im Verzeichnis */etc/libvirt/qemu* abgelegt. Neben der VM-Konfiguration werden auch die Netzwerkkonfigurationen von *libvirt* im XML-Format im Unterverzeichnis */etc/libvirt/qemu/networks* gespeichert.

Die XML-Datei einer VM besteht aus fest definierten Konventionen. Die XML-Datei der Beispiel-VM »Max« könnte so wie in Listing 20.4 aussehen:

```
<domain type='kvm'>
  <name>Max</name>
  <uuid>12345678-1234-1234-1234-123456789012</uuid>
  <memory>524288</memory>
  <currentMemory>524288</currentMemory>
  <vcpu>1</vcpu>
  <os>
  [...]
  </os>
  <features>
  [...]
  </features>
  <clock offset='utc' />
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <emulator>/usr/bin/kvm</emulator>
    <disk type='file' device='cdrom'>
    [...]
    </disk>
    <disk type='file' device='disk'>
      <source file='/var/lib/libvirt/images/Max.img' />
      <target dev='vda' bus='virtio' />
    </disk>
    <interface type='network'>
    [...]
    </interface>
    <graphics type='vnc' port='-1' autoport='yes' keymap='de' />
    <video>
    [...]
    </video>
  </devices>
</domain>
```

Listing 20.4 Beispiel-VM »Max.xml«

Die vorliegende Beispiel-VM »Max« ist eine kvm-VM mit dem Namen Max, 512 MB RAM und einer zugewiesenen CPU. Wie Sie Listing 20.4 entnehmen können, wird die gesamte Hardwarekonfiguration spezifiziert.

Die Netzwerkkonfigurationen werden in separaten XML-Dateien gespeichert. Eine NAT-Konfiguration sieht so wie in Listing 20.5 aus:

```
<network>
  <name>default</name>
  <uuid>12345678-1234-1234-12345678901234567</uuid>
  <forward mode='nat' />
  <bridge name='virbr0' stp='on' forwardDelay='0' />
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254' />
    </dhcp>
  </ip>
</network>
```

Listing 20.5 Beispiel-Netzwerkkonfiguration

Die Beispiel-Netzwerkkonfiguration verwendet den NAT-Modus und erzeugt die virtuelle Netzwerkschnittstelle `virbr0`. Darüber hinaus wird der DHCP-Dienst für das Netz `192.168.122.0/24` bereitgestellt. Den Clients werden dabei Adressen von 2 bis 254 zugewiesen.

20.5.2 Konsolenbasiertes Management: `virsh`

Das zur `libvirt` gehörige Konsolentool `virsh` ermöglicht das Anlegen, Verwalten und Administrieren virtueller Maschinen. In den Beispielen wurde eine lokale `KVM`-Umgebung verwendet. Das Programm `virsh` kann die Verbindung zu den VMs über unterschiedliche Verbindungstypen aufbauen, so wie alle auf `libvirt` basierten Managementlösungen. Die allgemeine Syntax von `virsh` lautet:

```
root@example:/# virsh COMMAND DOMAIN-ID [OPTIONS]
```

Listing 20.6 Syntax von »`virsh`«

Für `COMMAND` wird einer der in Tabelle 20.2 vorgestellten Befehle verwendet – eine vollständige Liste aller Befehle finden Sie in der Manpage. Mit `DOMAIN-ID` ist die interne numerische ID oder der Name des Gastes gemeint. In Abhängigkeit vom jeweiligen Befehl werden bestimmte Optionen mit `OPTIONS` angegeben. Testen Sie als Erstes die Verbindung zum *Dummy-Hypervisor*:

```
daniel@example:/# virsh -c test:///default list
Id  Name  State
-----
1   test  running
```

Listing 20.7 Verbindung zum Dummy-Hypervisor prüfen

Der *Dummy-Hypervisor* zeigt immer eine laufende VM an. Der Parameter `-c` gibt dabei an, dass sich *virsh* mit dem nachstehenden Hypervisor verbinden soll und nicht mit dem Default-Hypervisor. Der Parameter `list` gibt an, dass die zurzeit laufenden virtuellen Maschinen angezeigt werden sollen.

Die virtuellen Maschinen müssen bei *virsh* nicht zwingend auf dem lokalen System laufen. Es ist auch möglich, mehrere VM-Server zu betreiben und diese separat anzusteuern.



Irritierende Aufrufe: »qemu« und »kvm«

Lassen Sie sich nicht vom *virsh*-Aufruf irritieren. Sobald auf Ihrem System KVM aktiv ist, wird die Virtualisierungslösung auch verwendet. Weil QEMU und KVM dieselben Befehle nutzen (da KVM auf QEMU aufsetzt), wurde in *virsh* kein zweiter URI eingeführt.

Dafür können Sie die in Listing 20.8 dargestellte Syntax verwenden:

```
daniel@example:/# virsh -c qemu+ssh://root@vm01.example.com/system
```

Listing 20.8 Verbindung zu einem entfernten Host aufbauen

Dabei bauen Sie als Benutzer `root` mit dem Hypervisor `system` einen SSH-Tunnel zum System `vm01.example.com` auf.

Der nächste Test ist der Aufbau des Verbindungstyps `session`:

```
daniel@example:/# virsh -c qemu:///session list
Id      Name                               State
-----
```

Listing 20.9 Verbindung zu »session« prüfen



Der Verbindungstyp `session` zeigt alle virtuellen Maschinen, die Sie unter Ihrem lokalen Benutzerkonto gestartet haben. Dieser Verbindungstyp ist bei der Servervirtualisierung nicht zu empfehlen, da die virtuellen Maschinen mit weniger Rechten arbeiten und zum Beispiel bei einem Neustart des Wirt-Systems nicht automatisch gestartet werden können.

Abschließend wird der Verbindungstyp `system` getestet, der den gängigen Betriebsmodus für die Servervirtualisierung darstellt:

```
tom@example:/# virsh -c qemu:///system list
error: failed to connect to the hypervisor
error: Failed to connect socket to '/var/run/libvirt/libvirt-sock': Permission denied
```

Listing 20.10 Verbindung zu »system«

Dieser Verbindungstyp benötigt privilegierte Rechte, daher müssen Sie ihn als `root`-Benutzer ausführen. Da das Administrieren direkt als `root`-Benutzer nicht zu empfehlen ist, sollten Sie

die Konfiguration von *qemu* und *libvirt* anpassen, sodass Sie als unprivilegierter Benutzer mit *libvirt* arbeiten können. Standardmäßig wird der Installationsbenutzer bereits als Virtualisierungsadministrator eingerichtet. Beim Verbindungstyp *system* handelt es sich um den *default*-Hypervisor. Rufen Sie *virsh* ohne Angabe des Parameters *-c* auf, versucht *virsh* sich mit diesem Hypervisor zu verbinden.

Damit auch normale Benutzer mit *virsh* auf dem Verbindungstyp *system* arbeiten können, müssen Sie diese der Gruppe *libvirt* hinzufügen:

```
daniel@example:~# sudo usermod -a -G libvirt tom
```

Listing 20.11 Den Benutzer »tom« der Gruppe »libvirt« hinzufügen

Nach dem Neustart des *libvirt*-Daemons mit `systemctl restart libvirtd` können sich auch normale Benutzer über den Verbindungstyp *system* verbinden. Rufen Sie *virsh* ohne Parameter auf, so gelangen Sie in den Konsolenmodus. Dort können Sie alle Parameter direkt als Befehl absetzen:

```
daniel@example:~# virsh
Welcome to virsh, the virtualization interactive terminal.
```

```
Type: 'help' for help with commands
      'quit' to quit
```

```
virsh # list --all
 Id      Name                               State
-----
 -      Max                                ausschalten
 -      Moritz                             ausschalten
```

Listing 20.12 »virsh«-Konsolenmodus

Über den Befehl *help* erhalten Sie eine vollständige Liste aller zur Verfügung stehenden Befehle, die *virsh* verarbeiten kann. Die gängigsten Befehle von *virsh* finden Sie in Tabelle 20.2.

Befehl	Bedeutung
<code>list [all inactive]</code>	Zeigt eine Übersicht der laufenden, aller oder inaktiver virtueller Maschinen.
<code>start</code>	Startet eine VM, deren ID oder Name als Parameter übergeben wird.
<code>stop</code>	Stoppt eine VM.
<code>shutdown</code>	Führt eine VM kontrolliert herunter.

Tabelle 20.2 Auszug: »virsh«-Befehle und -Parameter

Befehl	Bedeutung
destroy	Bricht eine laufende VM ab.
define	Liest Änderungen an der Konfigurationsdatei einer VM ein.
undefine	Löscht eine VM.
suspend	Pausiert eine VM.
resume	Setzt eine pausierte VM fort.
reboot	Startet eine VM neu.
quit	Beendet den Konsolenmodus von <i>virsh</i> .

Tabelle 20.2 Auszug: »virsh«-Befehle und -Parameter (Forts.)

20.5.3 Virtuelle Maschinen installieren

So, wie es vielfältige Möglichkeiten der Netzwerkkonfiguration gibt, existieren für Virtualisierungslösungen auch viele Möglichkeiten, wie Sie die virtuellen Maschinen installieren. Wie bereits erörtert wurde, können Sie einzelne Systeme einfach über die Bordmittel der Virtualisierungslösungen einrichten. Bei größeren Umgebungen bieten grafisch orientierte Tools eine komfortablere Möglichkeit zur Administration. In diesem Abschnitt erfahren Sie, wie Sie schnell und erfolgreich Ihre virtuellen Maschinen einrichten können.

Direkt mit QEMU

Die Befehle von QEMU und KVM sind identisch. Dies wird beim Öffnen der Manpage von KVM deutlich, da dort auf die Manpage von QEMU verwiesen wird. Die Hardwarevirtualisierung wird von QEMU allerdings nicht genutzt. Direkt mit QEMU gestartete virtuelle Maschinen werden nur emuliert. Dies ist deutlich langsamer und daher nicht zu empfehlen. Vor der Installation einer virtuellen Maschine muss eine virtuelle Festplatte angelegt werden. Eine virtuelle Festplatte legen Sie mit QEMU wie folgt an:

```
root@example:/srv/virt# qemu-img create <SYSTEM NAME>.img 10G
```

Listing 20.13 Festplatte mit QEMU anlegen

Hierbei handelt es sich um einen Container fester Größe, der 10 GB groß ist. Um eine dynamisch wachsende Festplatte für die VM anzulegen, erweitern Sie den Befehl wie folgt:

```
root@example:/srv/virt# qemu-img create -f qcow2 <SYSTEM NAME>.img 10G
```

Listing 20.14 QEMU: dynamisch wachsende Festplatte anlegen

Der Parameter `-f` gibt das *Festplatten-Image-Format* (kurz FMT) an. Die Option `qcow2` wählt das aktuelle QEMU-eigene *Copy-on-Write-Format* in der Version 2. Hierbei handelt es sich um ein spezielles Format, das es dem Benutzer ermöglicht, *Overlay-Dateien* anzulegen. Diese beziehen sich auf ein vorhandenes Image. Änderungen werden nur in der Overlay-Datei gespeichert; das eigentliche Image, auch *Base-Image* genannt, bleibt unverändert.

So können leicht Template-Images erstellt werden. Zusätzlich ermöglicht das `qcow2`-Format das Erstellen von *Snapshots*⁶ einer VM. Diese werden direkt in der `qcow2`-Imagedatei gespeichert. Über Snapshots ist es möglich, Versionsstände einer VM anzulegen. So können Änderungen an einem virtuellen System einfach zurückgesetzt werden.

Das so angelegte Festplattenimage können Sie nun wie folgt mit KVM starten:

```
root@example:/srv/virt# kvm -hda Ubuntu-Server.img \
-cdrom /images/ubuntu-20.04-server-i386.iso -boot d
```

Listing 20.15 VM mit KVM starten

Über den Parameter `-hda` wird das Festplattenimage angegeben. Dem Parameter `-cdrom` wird das Image der Installations-CD oder das reelle CD-ROM-Laufwerk übergeben. Der abschließende Parameter `-boot` wird mit der Option `d` angewiesen, bei der Bootreihenfolge das CD-ROM-Laufwerk an die erste Stelle zu setzen. Tabelle 20.3 zeigt die Optionen, die der Parameter `-boot` verarbeiten kann. Ohne Angabe des Parameters `-boot` wird von der Festplatte gestartet. Nach dem Aufruf öffnet sich lokal ein Fenster mit der entsprechenden VM. Dort kann die Installation wie gewohnt durchgeführt werden.

Option	Erläuterung
a	Diskettenlaufwerk1
b	Diskettenlaufwerk2
c	Festplatte
d	CD-ROM
n-X	Netzwerkboot, wobei X die Adapternummer angibt (1–4 ist möglich)

Tabelle 20.3 KVM-Optionen: »boot«

Für virtuelle Maschinen, die direkt mit KVM installiert wurden, existiert keine Konfiguration. Lediglich das Festplattenimage ist vorhanden, sodass die Konfiguration jeweils mit dem Aufruf von `kvm` vorgenommen wird. Durch das Schließen des Fensters wird die VM beendet.

⁶ *Snapshot*, engl. für *Momentaufnahme*

Das QEMU-Fenster enthält noch eine Besonderheit. In ihm können Sie über die Tastenkombination `[Strg] + [Alt]`, gefolgt von den Tasten `[1]` bis `[4]`, zwischen unterschiedlichen QEMU-Konsolen wechseln. Tabelle 20.4 zeigt die jeweilige Belegung.

Konsole	Funktion
[1]	virtuelle Maschine
[2]	QEMU-Konsole
[3]	<i>serial0</i> -Schnittstelle
[4]	<i>parallel0</i> -Schnittstelle

Tabelle 20.4 »QEMU«-Konsolen

Über die QEMU-Konsole haben Sie die Möglichkeit, direkt Veränderungen an der VM vorzunehmen und sich Informationen zur VM ausgeben zu lassen.

20.5.4 virt-install

Eine weitere Möglichkeit zur Installation bietet das Tool *virt-install*, das sowohl KVM-basierte als auch Xen-basierte virtuelle Maschinen mit *libvirt* erzeugen kann. Das Programm gehört zur Familie der im Paket *virtinst* (CentOS: *virt-install*) enthaltenen Tools. Zur Installation einer VM setzen Sie folgenden Befehl ab:

```
root@example:/# virt-install --connect qemu:///system --hvm --name Fritz --ram 1024 \
--file /var/lib/libvirt/images/Fritz.img --file-size 5 --cdrom /images/debian.iso \
--vnc --os-variant debian11 --accelerate
```

Listing 20.16 VM-Installation mit »virt-install«

Im Gegensatz zur direkten Installation über *kvm* wird die virtuelle Maschine im Hintergrund gestartet und *virt-install* verbindet sich zur Ein- und Ausgabe nur mit der laufenden Instanz. Nach dem Absetzen des Befehls öffnet sich ein VNC-Fenster auf Ihrem System, in dem Sie die Installation des Betriebssystems wie gewohnt vornehmen können. Tabelle 20.5 zeigt die Bedeutung der einzelnen Parameter.

Parameter	Bedeutung
<code>--connect</code>	<i>libvirt</i> -Verbindungstyp
<code>--hvm</code>	Vollvirtualisierung (bei Xen -p für »Paravirtualisierung« angeben)

Tabelle 20.5 »virt-install«-Parameter

Parameter	Bedeutung
--name	Bezeichnung der VM
--ram	HauptspeichergroÙe in MB
--file	Speicherort der virtuellen Festplatte der VM
--file-size	Größe der Festplatte in GB
--cdrom	Pfad zum CD-ROM-Laufwerk oder zu einer ISO-Datei
--vnc	grafische Anbindung zur VM über VNC
--os-variant	exakte Angabe des Betriebssystems (Nachladen präziserer Parameter)
--accelerate	KVM-Unterstützung (ansonsten wird QEMU-VM gestartet)

Tabelle 20.5 »virt-install«-Parameter (Forts.)

Da keine gesonderte Angabe zur Netzwerkkonfiguration erfolgt ist, wird der Default geladen – das NAT. Bei der Installation über `virt-install` wird eine *libvirt*-XML-Datei für die virtuelle Maschine angelegt.

Für die soeben erstellte virtuelle Maschine *Fritz* wurde die Datei `/etc/libvirt/qemu/Fritz.xml` mit dem in Listing 20.17 dargestellten Inhalt angelegt:

```
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
    virsh edit Fritz
or other application using the libvirt API.
-->
<domain type='kvm'>
  <name>Fritz</name>
  <uuid>a827f94e-d52b-491d-badb-1b549ddcc6b4</uuid>
  <metadata>
    <libosinfo:libosinfo xmlns:libosinfo=\
      "http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://debian.org/debian/11"/>
    </libosinfo:libosinfo>
  </metadata>
  <memory unit='KiB'>1048576</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
  <vcpu placement='static'>2</vcpu>
  <os>
```

```
<type arch='x86_64' machine='pc-q35-6.2'>hvm</type>
<boot dev='hd' />
</os>
<features>
  <acpi />
  <apic />
</features>
<cpu mode='host-model' check='partial' />
<clock offset='utc'>
  <timer name='rtc' tickpolicy='catchup' />
  <timer name='pit' tickpolicy='delay' />
  <timer name='hpet' present='no' />
</clock>
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>destroy</on_crash>
<pm>
  <suspend-to-mem enabled='no' />
  <suspend-to-disk enabled='no' />
</pm>
<devices>
  <emulator>/usr/bin/qemu-system-x86_64</emulator>
  <disk type='file' device='disk'>
    <driver name='qemu' type='qcow2' discard='unmap' />
    [...]
  </disk>
</devices>
</domain>
```

Listing 20.17 Inhalt von »/etc/libvirt/qemu/Fritz.xml«

Alle Parameter der VM sind in der XML-Datei entsprechend aufgeführt. Selbstverständlich können Sie diese Parameter von Hand anpassen. Anschließend müssen Sie die Änderungen *libvirt* mitteilen. Führen Sie dazu den folgenden Befehl aus:

```
root@example:/# virsh define /etc/libvirt/qemu/Fritz.xml
Connecting to uri: qemu:///system
Domain Fritz von /etc/libvirt/qemu/Fritz.xml definiert
```

Listing 20.18 Änderungen mit »virsh« aktivieren

Wie in dem Hinweistext aus Listing 20.17 aber bereits erläutert wurde, sollten Sie Änderungen so nicht vornehmen, sondern das Programm `virsh edit <VM>` verwenden. Ihre Änderungen werden beim nächsten Start der virtuellen Maschine wirksam. Da *virt-install* auf *libvirt* aufsetzt, empfiehlt es sich, die Standardverzeichnisse zu verwenden. Legen Sie daher

die virtuellen Festplatten im Verzeichnis `/var/lib/libvirt/images/` ab. Die VM-Konfigurationsdatei wird von `virt-install` automatisch im Verzeichnis `/etc/libvirt/qemu/` abgelegt. Anders als beim direkten Start einer VM über KVM wird die VM durch das Schließen des VNC-Fensters nicht gestoppt. Sie läuft einfach im Hintergrund weiter. Um das Fenster erneut zu öffnen, können Sie den `virt-viewer` einsetzen (siehe Listing 20.19):

```
daniel@example:/# virt-viewer Fritz
```

Listing 20.19 Administrationsfenster einer laufenden VM öffnen

20.5.5 Alleskönner: Der Virtual Machine Manager

Das von Red Hat entwickelte Programm *Virtual Machine Manager (VMM)* stellt eine GUI zur Administration von virtuellen Maschinen bereit. Über die GUI können Sie den gesamten Funktionsumfang von `libvirt` nutzen. Über `virt-install` angelegte virtuelle Maschinen können Sie mit dem Virtual Machine Manager administrieren. Da dieses Programm über `libvirt` arbeitet, ist keine weitere Konfiguration notwendig (eine Desktopumgebung vorausgesetzt).

Installieren Sie das Paket `virt-manager` aus den Paketquellen Ihrer Distribution. Wie bei allen `libvirt`-basierten Programmen müssen Sie sich zunächst mit dem verwendeten Hypervisor verbinden. Dieser kann lokal oder auf einem entfernten System laufen. Stellen Sie nach dem Start des Programms `virt-manager` über das Menü DATEI • VERBINDUNG HINZUFÜGEN... eine Verbindung zum Hypervisor her. Nach der Herstellung der Verbindung lädt der Virtual Machine Manager die bestehende Konfiguration und stellt Ihnen diese übersichtlich dar. Im Startbildschirm erhalten Sie einige Informationen zu den bereits angelegten Systemen (siehe Abbildung 20.7). Diese können Sie dort ebenfalls starten oder stoppen. Sie können hier auch in den Konfigurationsbildschirm wechseln.

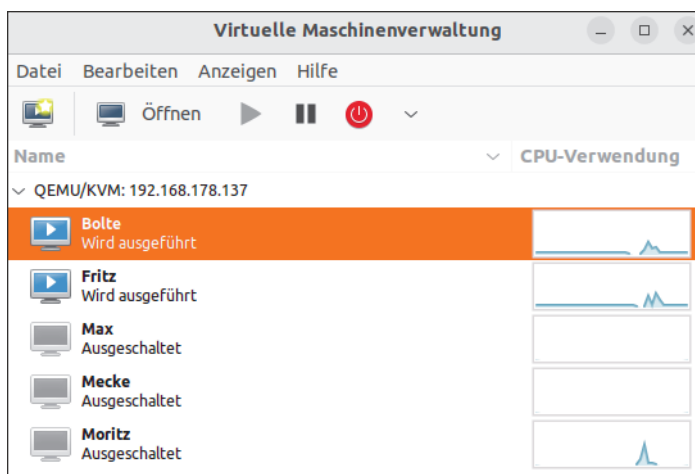


Abbildung 20.7 Startbildschirm »VMM«

Die Übersicht enthält die wichtigsten Informationen zum Hypervisor und zu den virtuellen Maschinen. Ebenfalls können Sie weitere virtuelle Maschinen anlegen. Legen Sie eine neue virtuelle Maschine über den Button NEU an. Anschließend erscheint der Dialog NEW VM, der Sie in fünf Schritten durch die Erstellung einer neuen VM führt. Folgende Schritte werden dabei durchlaufen:

1. Name und Installationsart

Hier geben Sie der neuen VM einen Namen und wählen die Installationsart aus. Sie können dabei zwischen folgenden Arten wählen:

- lokale Installation (ISO-Image oder CD-ROM)
- Netzwerkinstallation (*HTTP*, *FTP* oder *NFS*⁷)
- Netzwerkstart über PXE-Boot

2. Auswahl des Installationsmediums und des Betriebssystems

Entsprechend Ihrer vorherigen Auswahl können Sie hier das CD-ROM-Laufwerk, das ISO-Image oder die URL zur Netzwerkinstallation angeben. Falls Sie vorher *PXE-Boot* ausgewählt haben, entfällt diese Auswahl. Zusätzlich können Sie das Betriebssystem spezifizieren, das die VM erhalten soll. Geben Sie so genau wie möglich das Betriebssystem an, damit *libvirt* vordefinierte Parameter setzen und somit die VM verbessern kann.

3. CPU und RAM

Hier können Sie der VM Prozessoren und Hauptspeicher zuweisen.

4. Virtuelle Festplatte

Wählen Sie hier die virtuelle Festplattengröße und die Art (statisch oder wachsend). Zusätzlich besteht die Möglichkeit, bereits vorhandene virtuelle Festplatten der neuen VM zuzuweisen. Somit können Sie vorhandene Installationen, die zum Beispiel direkt mit KVM vorgenommen wurden, in den Virtual Machine Manager übernehmen.

5. Übersicht und erweiterte Einstellungen

Abschließend erhalten Sie eine Übersicht über Ihre Konfiguration. Über das Menü ADVANCED OPTIONS können Sie der Netzwerkschnittstelle eine feste MAC-Adresse geben, den Netzwerktyp definieren sowie den Typ der Virtualisierung (Hypervisor) und die VM-Architektur festlegen.

Anschließend wird die neue VM Ihrer Konfiguration entsprechend angelegt und gestartet. Die Installation erfolgt in einem separaten Fenster. In diesem Fenster können Sie sich die Ausgabe der VM im Reiter KONSOLE anzeigen lassen. Die gegenwärtig zugewiesene Hardware wird im Reiter DETAILS dargestellt. Über den Button GERÄT HINZUFÜGEN können Sie der VM zusätzliche Hardware zur Verfügung stellen. Dazu zählen SPEICHER für Festplatten (virtuelle Festplatten oder ganze physische Partitionen), NETZWERK für Netzwerkschnittstellen, EINGABE für Eingabegeräte, GRAFIK für die grafische Anbindung zur VM, KLANG für Audiogeräte

⁷ *Network File System*, engl. für *Netzwerkdateisystem* – stellt Verzeichnisse im Netzwerk zur Verfügung.

und PCI HOST-GERÄT für etwaige PCI- und USB-Geräte. Fügen Sie der VM eine weitere Netzwerkkarte hinzu, indem Sie im Dialog NEUE VIRTUELLE GERÄTE HINZUFÜGEN zunächst als TYP den Punkt NETZWERK auswählen. Im rechten Teil des Fensters können Sie die Art der Netzwerkanbindung wählen (siehe Abbildung 20.8).

Durch Auswahl der NETZWERKQUELLE können Sie die virtuelle Netzwerkkarte über die vorher angelegte Netzwerkbrücke *br0* direkt mit dem physischen Netzwerk verbinden. Ebenso stehen Ihnen virtuelle Netzwerke zur Verfügung. Da noch keine virtuellen Netzwerke definiert wurden, wird Ihnen hier lediglich das *Default*-Netzwerk angeboten, das das Netzsegment 192.168.122.0/24 im NAT-Modus zur Verfügung stellt. Zusätzlich können Sie der virtuellen Netzwerkkarte eine feste MAC-Adresse zuweisen. Als letzter Auswahlpunkt steht das Modell der Netzwerkkarte zur Wahl. Nicht jedes Betriebssystem arbeitet mit jedem Netzwerkkartenmodell fehlerfrei zusammen. Falls Ihr System *virtio* unterstützt, sollten Sie dies als Modell verwenden, um die größtmögliche Performance zu erreichen.

Abschließend erhalten Sie eine Übersicht über die Konfiguration und können die neue virtuelle Hardware über den Button FERTIG anlegen.

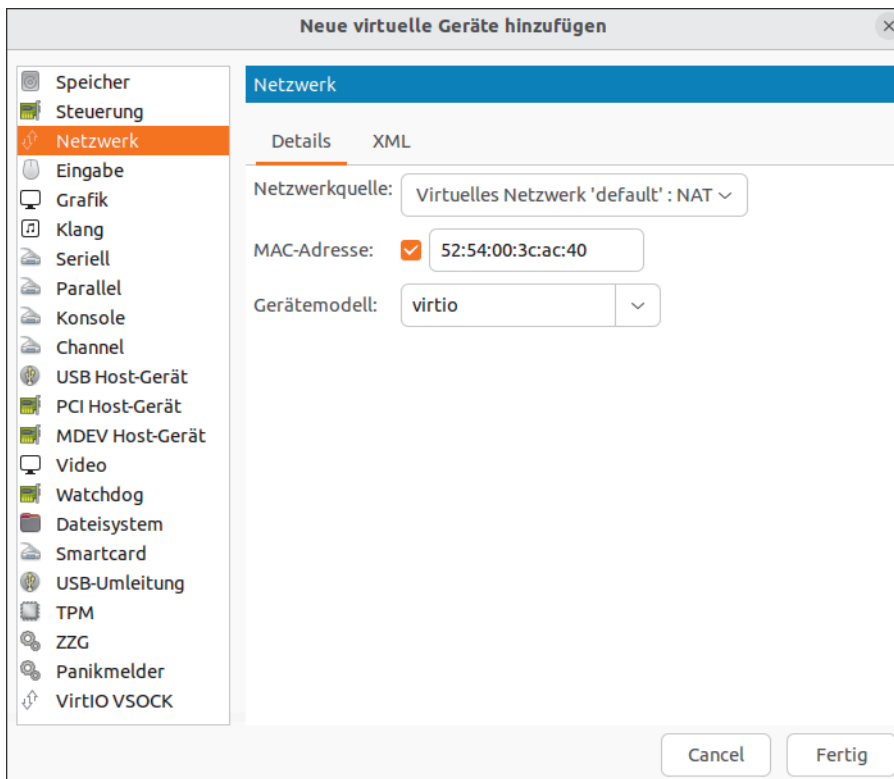


Abbildung 20.8 Neue virtuelle Netzwerkkarte einrichten

Damit Sie eine eigene DMZ-Struktur für Ihre virtuellen Maschinen betreiben können, müssen Sie diese vorab definieren. Öffnen Sie dafür aus dem Startbildschirm des Virtual Machine Managers mit einem Rechtsklick auf den jeweiligen Hypervisor (im Beispiel: LOCALHOST (SYSTEM)) das Menü DETAILS. Das sich nun öffnende Fenster VERBINDUNGSDetails enthält im Reiter ÜBERSICHT alle Informationen zum Hypervisor. Im Reiter VIRTUELLES NETZWERK erhalten Sie eine Übersicht über die für diesen Hypervisor angelegten virtuellen Netzwerke. Fügen Sie ein neues virtuelles Netzwerksegment über den Button + hinzu. Der anschließende Dialog ERSTELLE VIRTUELLES NETZWERK führt Sie in vier Schritten zum Ziel:

► **Name**

Vergeben Sie hier den Namen des Netzwerks. Dieser wird bei der Auswahl in den VM-Details angezeigt.

► **Modus**

Wählen Sie, ob es sich bei dem virtuellen Netzwerk um ein isoliertes Netzwerk oder um ein an das physische Netz weitergeleitetes Netzwerk oder um ein NAT handelt. Bei den beiden letzten können Sie noch spezifizieren, auf welche physische Netzwerkschnittstelle der Modus angewandt werden soll.

► **IPv4 Konfiguration – Netzwerksegment und DHCP-Adressraum**

Legen Sie hier das Netzwerksegment fest, und stellen Sie den Raum für die Adressvergabe via DHCP ein. Standardmäßig wird Ihnen immer die Hälfte des vorher spezifizierten Netzwerksegments angeboten.

► **IPv6**

Dies ist standardmäßig deaktiviert. Falls Sie aber IPv6 mit Ihren Gästen verwenden wollen, müssen Sie diese Option hier aktivieren und konfigurieren.

► **DNS Domänenname**

Hier können Sie eine abweichende DNS-Domäne angeben oder die vom System verwenden lassen.

Ab sofort steht Ihnen das neu angelegte virtuelle Netzwerk in den DETAILS der virtuellen Maschinen zur Verfügung. Über den Reiter STORAGE können Sie die Lagerorte der virtuellen Festplatten einsehen und konfigurieren.

Über den Button + können Sie ebenfalls weitere Lagerorte hinzufügen. Diese können aus lokalen Dateien, lokalen Partitionen, lokalen Festplatten, *iSCSI*⁸-Zielen, LVM-Gruppen oder NFS-Pfaden bestehen.

Die Auslagerung der virtuellen Festplatten kommt vor allem bei der Clusterbildung von Virtualisierungslösungen zum Tragen, da dort die virtuellen Maschinen beim Ausfall oder zu Wartungszwecken auf einen anderen Server übertragen werden müssen.

8 *internet Small Computer System Interface*: SCSI-Protokoll über TCP. Es dient zur Anbindung entfernter Festplatten an das eigene System, so als wären diese über den lokalen SCSI-Bus verbunden.

20.5.6 Zusätzliche Konsolentools

Die zusätzlichen Fähigkeiten des Virtual Machine Managers (wie die Auslastungsanzeige) können Sie selbstverständlich auch über die Konsole abrufen. In diesem Abschnitt stellen wir Ihnen die gängigsten Tools zur Administration von VMs auf der Konsole vor.

Auslastungsanzeige: `virt-top`

Mit `virt-top` erhalten Sie eine Auslastungsanzeige der virtuellen Maschinen im `top`-Stil. Nach der Installation des Programms aus den Paketquellen können Sie es direkt mit `virt-top` aufrufen:

```
virt-top 18:36:51 - 2/2CPU 2294MHz 4938MB 3,1%
2 domains, 1 active, 1 running, 0 sleeping, 0 paused, 1 inactive D:0 O:0 X:0
CPU: 41,6% Mem: 2048 MB (2048 MB by guests)

  ID S RDRQ WRRQ RXBY TXBY %CPU %MEM  TIME  NAME
   1 R   0   0   0   0 41,6 20,0 1:11.29 Moritz
   -                                     (Max)
```

Listing 20.20 Ausgabe von »`virt-top`«

Über den Parameter `-c` können Sie wie bei allen `libvirt`-basierten Tools den Hypervisor angeben, mit dem sich das Tool verbinden soll.

Klonen von virtuellen Maschinen: `virt-clone`

Mit diesem Tool ist das einfache Vervielfältigen von virtuellen Maschinen möglich. Das von Kazuki Mizushima entwickelte Tool ist Bestandteil des Pakets `virtinst` (CentOS: `virt-install`). Der Befehl aus Listing 20.21 erstellt einen Klon *Mecke* von der vorhandenen VM *Moritz*:

```
root@example:/# virt-clone -o Moritz -n Mecke -f /var/lib/libvirt/images/Mecke.img
Zuweisen von 'Mecke.img' 39% [===== ] 110 MB/s | 2.0 GB 00:00:27 ETA
```

Listing 20.21 Vorhandene VM »*Moritz*« nach »*Mecke*« klonen

Das Programm `virt-clone` kopiert nun die virtuelle Festplatte und die entsprechende Konfiguration, die direkt umbenannt wird. Über den Parameter `-o` referenzieren Sie das Original. Mit `-n` geben Sie den Namen des Klons an. Den Speicherort der neuen virtuellen Festplatte geben Sie mit dem Parameter `-f` an.

Augen auf: `virt-viewer`

Das von Daniel P. Berrange geschriebene Programm `virt-viewer` stellt eine VNC-Verbindung zu einer laufenden virtuellen Maschine her. Nach der Installation des Pakets `virt-viewer` können Sie in der gewohnten `libvirt`-Syntax eine Verbindung zum Hypervisor und zu der darauf laufenden VM aufbauen:

```
daniel@example.net:~# virt-viewer Moritz
```

Listing 20.22 Aufruf von »virt-viewer«

Anschließend öffnet sich ein VNC-Fenster zur Administration der angegebenen VM. Über den optionalen Parameter `-c` können Sie einen Hypervisor angeben. Der optionale Parameter `-d` baut eine direkte VNC-Verbindung zur VM auf, auch wenn die Hypervisor-Anbindung über einen SSH-Tunnel oder TCP-Socket läuft.

20.6 Umzugsunternehmen: Live Migration

Die *Live Migration* ist eine Funktion, bei der ein Gast-System im laufenden Betrieb ohne Ausfall auf ein anderes Wirt-System verschoben wird. Dabei sollten beide Wirt-Systeme möglichst über die gleiche Architektur verfügen. Dies ist aber nicht zwingend notwendig. Die Migration von *AMD* auf *Intel* ist ebenso möglich wie die Migration von Linux- auf Windows-Wirt-Systeme. Da bei einer Live Migration die virtuelle Festplatte nicht erst kopiert wird, ist der Einsatz eines Netzwerkdateisystems wie *NFS*, *iSCSI* oder – was die effektivste Möglichkeit bietet – über ein *SAN*⁹ zwingend erforderlich. Ebenso müssen beide Wirt-Systeme über die gleiche Netzwerkanbindung verfügen. Der Virtual Machine Manager bietet Ihnen dabei alle Konfigurationsmöglichkeiten, um eine Live Migration umzusetzen.

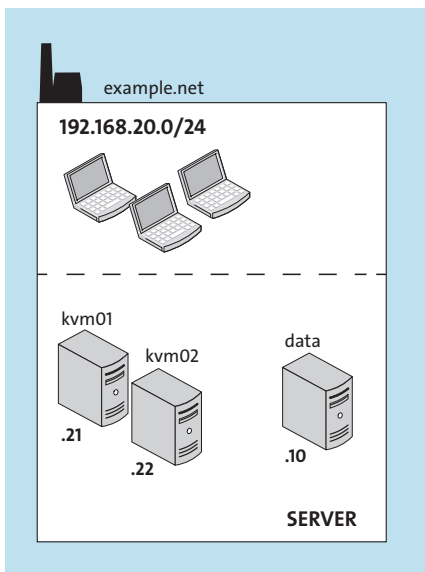


Abbildung 20.9 Netzwerkkonfiguration für eine »Live Migration«

⁹ *Storage Area Network*, engl. für *Speichernetzwerk*, ein autarkes Hochgeschwindigkeitsnetzwerk zur Anbindung an einen Datenspeicher

In diesem Abschnitt wird eine NFS-Freigabe auf dem System *data* eingerichtet. Die beiden KVM-Server *kvm01* und *kvm02* binden diese ein. Auf den KVM-Servern wird anschließend im Virtual Machine Manager die Konfiguration für die Live Migration implementiert. Abbildung 20.9 stellt die Netzwerkkonfiguration der Testumgebung dar.

20.6.1 Vorbereitungen

Damit Ihre virtuellen Maschinen im laufenden Betrieb verschoben werden können, müssen Sie zunächst einen Netzwerkspeicher einrichten. Dazu verwenden Sie das altbekannte NFS. Installieren Sie auf dem System, das den Datenspeicher zur Verfügung stellen soll (im Beispiel *data*), das Paket *nfs-kernel-server*.

Anschließend können Sie in der zentralen Konfigurationsdatei */etc/exports* die Netzwerkfreigaben definieren:

```
/data          10.1.0.0/255.255.255.0(rw,async,no_subtree_check)
```

Listing 20.23 NFS-Freigabe »data« in »/etc/exports«

In Listing 20.23 wurde das Verzeichnis */data/* dazu auserkoren, die virtuellen Festplatten zu beinhalten. Die Freigabe darf aus dem Server-Netzwerk 10.1.0.0/24 geöffnet werden. Die neue Freigabe muss aktiviert werden, damit sie eingebunden werden kann.

Starten Sie dafür den *nfs-kernel-server* über dessen *init*-Skript neu. Alternativ setzen Sie den folgenden Befehl ab:

```
root@data:/# exportfs -ra
```

Listing 20.24 Exportieren der Freigabe »/data«

Der Befehl *exportfs* lädt die Freigaben aus der Datei */etc/exports* neu. Die Parameter *-r* und *-a* veranlassen dabei, dass die Laufzeitvariablen und *-dateien* entsprechend den neuen Gegebenheiten gesetzt werden.

Auf der Clientseite muss das Paket *nfs-common* installiert sein. Damit ist es möglich, dem *mount*-Befehl eine *NFS-URL* zu übergeben und so die Netzwerkfreigabe in das eigene System zu integrieren. Setzen Sie zum Einbinden der Freigabe */data* den in Listing 20.25 dargestellten Befehl ab, und wiederholen Sie diesen auf dem zweiten KVM-Server:

```
root@kvm01:/# mount -t nfs 10.1.0.0:/data /data
```

Listing 20.25 Einbinden der Freigabe »/data«

20.6.2 Konfiguration im Virtual Machine Manager

Verbinden Sie zunächst beide Virtual Machine Manager mit dem jeweils anderen Hypervisor. Anschließend legen Sie auf beiden Wirt-Systemen ein weiteres *Storage* an. Öffnen Sie dazu

im Virtual Machine Manager die VERBINDUNGSDetails beider Wirt-Systeme, und richten Sie im Reiter SPEICHER über die +Schaltfläche einen neuen Speicher-Pool ein. In dem Dialog NEUEN SPEICHER-POOL HINZUFÜGEN geben Sie dem neuen Datenspeicher zunächst einen Namen und wählen als Typ NETFS: NETZWERK EXPORTIERTES VERZEICHNIS aus.

Abbildung 20.10 NFS-Freigabeparameter

Über die Schaltfläche FERTIG erstellen Sie die NFS-Freigabe (siehe Abbildung 20.10). Richten Sie die Freigabe wie in der Abbildung gezeigt ein. Anschließend installieren Sie eine neue virtuelle Maschine auf einem der Wirt-Systeme.

Nach der Installation können Sie die angelegte VM über einen Rechtsklick im Startbildschirm des Virtual Machine Managers und durch Auswahl des Punkts MIGRATE auf den anderen KVM-Server migrieren. Die virtuelle Maschine wird nun auf den zweiten KVM-Server verschoben.



Netzwerkconfiguration der VM

Sollte die virtuelle Maschine in einem virtuellen Netzwerk angebunden sein, muss auf dem Ziel-Wirt-System diese Konfiguration ebenfalls existieren. Werden die KVM-Server als Master/Backup-Systeme betrieben, ist es sinnvoll, die virtuelle Netzwerkconfiguration zu synchronisieren. Kopieren Sie dafür die XML-Dateien aus dem Verzeichnis `/etc/libvirt/qemu/networks/` auf den jeweils anderen KVM-Server.

Kapitel 21

Containervirtualisierung mit Docker und Podman

Seit einigen Jahren sind Software-Container und die zugehörigen Ökosysteme aus der IT-Welt nicht mehr wegzudenken. Die leichtgewichtige Container-Virtualisierung kann an vielen Stellen ihre Vorteile ausspielen. In diesem Kapitel erlernen Sie die Basics im Umgang mit Docker und Podman.

21.1 Einführung, Installation und Grundlagen für den Betrieb

21.1.1 Was ist ein Container?

Der Begriff des (Software-)Containers ist nicht ganz leicht zu fassen. Versuchen wir es mit folgender Definition:

Ein Container ist eine standardisierte Software-Einheit, mit der eine Applikation isoliert von ihrer Umgebung betrieben werden kann.

Eine etwas pragmatischere Betrachtung des Themas könnte zu zwei Sichtweisen bzw. Standpunkten führen:

Der High-Level-Standpunkt: Ein Container ist eine leichtgewichtige virtuelle Maschine

- ▶ Ein Container hat ein eigenes Dateisystem, eigene Prozesse, eigene Netzwerkschnittstellen etc.
- ▶ Sie können sich (in der Regel) mit einem Container verbinden und darin in einer Shell arbeiten.
- ▶ Sie könnten in einem Container Programme starten und Software installieren.

Der Low-Level-Standpunkt: Ein Container ist nur ein aufgemotztes chroot

- ▶ Ein Container hat keinen eigenen Kernel, sondern nutzt den Kernel des Hostsystems.
- ▶ Ein Container hat bzw. braucht kein Init-System wie systemd.
- ▶ Ein Container hat bzw. braucht keine Standarddienste wie Syslog, Cron, SSH usw.
- ▶ Letztlich besteht ein Container nur aus einem Haufen Dateien und einem Haufen Prozesse; Letztere können Sie sogar einfach von der Hostmaschine aus sehen.

Beide Standpunkte sind in gewisser Weise »richtig«. Die technischen Low-Level-Aspekte werden wir später auch noch genauer unter die Lupe nehmen (sofern es für das Verständnis wichtig ist).

21.1.2 Container vs. VM

Oft hört man, dass Container in Konkurrenz zu unseren altbewährten virtuellen Maschinen (VMs) stehen. Es ist jedoch nicht zu erwarten, dass man irgendwann keine klassischen VMs mehr brauchen wird, denn Container und VMs sind Lösungen für zumeist sehr unterschiedliche Problemstellungen. Jemand hat es mal ziemlich gut auf den Punkt gebracht:

Container sind Prozesse, VMs sind Server.

Nichtsdestotrotz gibt es sicher Situationen, in denen Sie mit einem leichtgewichtigen Container besser bedient wären als mit einer »klobigen« VM. Das muss von Fall zu Fall entschieden werden – eine Lösung für alles sind weder VMs noch Container. Zurzeit werden beide Technologien jedenfalls parallel eingesetzt und ergänzen sich dabei ausgezeichnet: VMs, in denen Container laufen! Technische Unterschiede (ohne Anspruch auf Vollständigkeit) sind:

- ▶ VMs sind Server mit virtueller Hardware, BIOS und komplettem Betriebssystem. Container haben (und brauchen) all dies nicht.
- ▶ Mit Containern erreichen Sie eine höhere Packungsdichte, zumindest was Plattenplatz angeht. CPU, RAM und Netzwerk stehen jedoch auch hier nur in endlichem Maß zur Verfügung.
- ▶ In Sachen Security müssen Container aufgrund der geringeren Abschottung vom Hostsystem theoretisch eher als problematisch eingestuft werden. In der Praxis haben sich bislang allerdings keine ernsthaften Sicherheitsprobleme gezeigt.
- ▶ Bei der Portabilität haben VMs immer noch die Nase vorn. Einen für Linux erstellten Container können Sie nicht »einfach so« unter Windows laufen lassen (und umgekehrt). Auch unterschiedliche Prozessor-Architekturen wie x86 vs. ARM sind nicht überbrückbar (obwohl Linux und Docker bzw. Podman auf beiden laufen). Das ist zugegebenermaßen aber auch mit VMs schwierig.

21.1.3 Entstehung und Geschichte

Docker

Die Open-Source-Software *Docker* wurde Anfang 2013 erstmals veröffentlicht. Der ursprüngliche Name der Entwicklerfirma war »dotCloud, Inc.«; gegen Ende 2013 wurde die Firma umbenannt in »Docker, Inc.«. Im Jahr 2014 entstand ein regelrechter Hype um das Thema »Container«, und für die meisten ist der Begriff bis heute synonym mit »Docker-Container«.

Und die Firma Docker freute sich über Investitionen von weit über 100 Millionen US-Dollar. Leider hat Docker, Inc. es letztlich nicht geschafft, mit seiner Plattform und seinem guten Namen auch dauerhafte finanzielle Erfolge zu erzielen. Ende 2019 wurde die Enterprise-Sparte von Docker vom Cloud-Dienstleister Mirantis übernommen; Docker, Inc. kümmert sich nun verstärkt um seine Produkte *Docker Engine*, *Docker Desktop* und *Docker Hub*.

Podman

Hinter Podman steht die Firma Red Hat. Man war dort mit gewissen technischen Aspekten der Docker-Software unzufrieden und begann 2017 mit der Entwicklung einer Alternative, mit der u. a. eine bessere Kompatibilität mit Kubernetes erreicht werden konnte. (Nicht zuletzt wollte man sicher auch der Marktmacht von Docker etwas entgegensetzen.)

Im Jahr 2019 wurde die Podman-Version 1.0 veröffentlicht. Sie fand zumindest in der Red-Hat-Welt schnell Verbreitung, da das Podman-Toolset auch erstmalig in der damals neuen Version 8 von RHEL enthalten war. Dabei war und ist es sicher auch entscheidend, dass Podman zwar intern viele Dinge anders angeht und löst, nach außen aber weitestgehend aufrufkompatibel mit Docker ist, sodass Docker-erfahrene Admins sich nicht völlig neu einarbeiten mussten bzw. müssen.

Container – eigentlich ein alter Hut?

Das Thema Container ist in der IT-Welt alles andere als neu. Bereits in den 1960er-Jahren fand sich auf Großrechnern in technologischer Hinsicht Vergleichbares. Durchaus ähnlich, aber etwas neueren Datums sind auch die folgenden Technologien:

- ▶ UNIX chroot (~1979)
- ▶ FreeBSD Jails (~2000)
- ▶ Solaris Containers/Zones (~2005)
- ▶ OpenVZ (~2005)
- ▶ Linux-VServer (~2005)
- ▶ LXC (Linux Containers, ~2008)

Docker setzte anfänglich sogar noch auf der LXC-Schnittstelle des Linux-Kernels auf; später wurde dann eine eigene Schnittstelle namens `libcontainer` entwickelt.

Und warum kam ausgerechnet mit Docker der große Durchbruch? Wahrscheinlich war es die Mischung aus anwenderfreundlichem Toolset und »Die Zeit war einfach reif dafür«.

21.1.4 Versionen

Im Mittelpunkt unserer Betrachtungen steht das Produkt *Docker Engine* (Details unter der URL <https://docs.docker.com/engine/>), eine freie Client-Server-Anwendung, die ausschließ-

lich auf Linux-Plattformen eingesetzt werden kann. Windows- bzw. Mac-User greifen zum Produkt *Docker Desktop*, das mit einem anderen Lizenzmodell vertrieben wird. Früher hieß die Docker Engine »*Docker CE*« (= Community Edition). Dieser alte Name ist auch heute noch gebräuchlich und findet sich beispielsweise bei der Benennung von Distributionspaketen.

Die Versionierung folgt seit 2017 einem Datumsschema. »17.12.x« z. B. ist die Version vom Dezember 2017. (Etwas genauer gesagt, geht es hier immer um das Datum des *geplanten* ersten Erscheinungstermins, nicht um das tatsächliche Erscheinungsdatum.) Wenn Sie die Release-Notes unter die Lupe nehmen möchten: <https://docs.docker.com/engine/release-notes/>.

Podman (<https://podman.io>) folgt einem gewöhnlichen semantischen Versionierungsschema. Informationen zu Podman-Releases finden Sie unter <https://podman.io/releases>.

21.1.5 Docker oder Podman?

Falls Sie es sich wirklich aussuchen können, weil Ihr Vorhaben bzw. Projekt diesbezüglich keine Vorgaben macht: Sie machen mit keinem der beiden Produkte etwas falsch. Docker und Podman unterscheiden sich allerdings deutlich in der Art der Implementierung.

Der wahrscheinlich größte Unterschied ist, dass Docker einen zentralen Daemon benötigt, wohingegen Podman »Daemon-less« arbeitet. Eine zentrale Instanz, die alle Container überwacht und diese beispielsweise bei Bedarf neu startet, muss aber kein Nachteil sein. Dass diese Instanz in der Vergangenheit stets mit Rootrechten laufen musste, war schon eher ein berechtigter Kritikpunkt. Seit 2019 gibt es zwar auch »Rootless Docker«, was aber wegen eines gewissen Konfigurationsaufwands oft nicht genutzt wird. Ein offensichtlicher Vorteil von Podman, der auch mit Rechten zu tun hat, ist der folgende: Sie können mit Podman auch als unprivilegierter User sofort Container starten. In einem Docker-Standard-Setup hingegen müssen Sie dazu entweder Root sein oder Mitglied in der Gruppe `docker` (wodurch Sie sich in der Praxis aber auch sehr schnell Root-Rechte beschaffen könnten.)

alias docker=podman

In unserer kleinen Einführung wollen wir nun folgenden Ansatz verwenden: Im Mittelpunkt unserer Betrachtungen soll nach wie vor das »klassische« Docker stehen. Wenn Sie ein Podman-basiertes Setup bevorzugen, setzen wir voraus, dass Sie folgenden Alias gesetzt haben (z. B. in Ihrer `~/.bashrc`):

```
alias docker=podman
```

Das ist in der Tat eine von den Podman-Entwicklern empfohlene Vorgehensweise, wie Sie vielleicht schon auf der Projektseite gesehen haben (eines der Designziele von Podman war ja die Aufrufkompatibilität zu Docker). Es wird aber sicher Situationen geben, in denen die technischen Unterschiede eine solch einfache Vorgehensweise nicht zulassen. Dort werden wir dann gesondert auf beide Produkte eingehen.

Warum nicht beides?

Es sollte problemlos möglich sein, beide Produkte parallel auf demselben Host zu betreiben. Wir weisen aber ausdrücklich darauf hin, dass wir ein solches Setup nicht in allen Aspekten getestet haben, und würden in produktiven Umgebungen eher davon abraten. Entscheiden Sie sich für eines der beiden Werkzeuge.

21.1.6 Installation von Docker

Wir beschreiben nun die Schritte, mit denen Sie unter den gängigen Linux-Distributionen an eine aktuelle Docker-Engine-Installation gelangen. In fast allen Fällen nutzen wir dazu die von Docker offiziell zur Verfügung gestellten Repositories, da die Distributionen die Software teilweise gar nicht bzw. nur in veralteten Versionen an Bord haben.

Debian 11

Schritt für Schritt: Installation von Docker Engine unter Debian 11



1 Installieren Sie grundlegende Pakete:

```
# apt update
# apt install apt-transport-https ca-certificates gnupg2 wget
```

2 Importieren Sie den offiziellen Docker-Key:

```
# wget -qO - https://download.docker.com/linux/debian/gpg | apt-key add -
```

3 Prüfen Sie gegebenenfalls den Fingerprint:

```
# apt-key fingerprint docker
[...]
9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88
[...]
```

4 Fügen Sie das Docker-Repository als Paketquelle hinzu:

```
# echo deb https://download.docker.com/linux/debian \
$(lsb_release -cs) stable >/etc/apt/sources.list.d/docker-ce.list
```

```
# apt update
```

5 Installieren Sie Docker CE:

```
# apt install docker-ce
```



Ubuntu 22.04 LTS



Schritt für Schritt: Installation von Docker Engine unter Ubuntu 22.04 LTS

1 Installieren Sie grundlegende Pakete:

```
$ sudo apt-get update
$ sudo apt-get install apt-transport-https ca-certificates wget
```

2 Importieren Sie den offiziellen Docker-Key:

```
$ wget -qO - https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

3 Prüfen Sie gegebenenfalls den Fingerprint:

```
$ sudo apt-key fingerprint docker
[...]
          9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
[...]
```

4 Fügen Sie das Docker-Repository als Paketquelle hinzu:

```
$ sudo add-apt-repository "deb [arch=amd64] \
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

```
$ sudo apt-get update
```

(Die Architektur ist gegebenenfalls anzupassen, z.B. auf armhf.)

5 Installieren Sie Docker CE:

```
$ sudo apt-get install docker-ce ■
```

Rocky Linux 9, CentOS Stream 9



Schritt für Schritt: Installation von Docker Engine unter Rocky Linux 9 / CentOS Stream 9

1 Fügen Sie das Docker-CE-Repository als Paketquelle hinzu:

```
# dnf config-manager \
--add-repo https://download.docker.com/linux/centos/docker-ce.repo
```

2 Installieren Sie Docker CE:

```
# dnf install docker-ce
```

(Der Key-Fingerprint ist 060A 61C5 1B55 8A7F 742B 77AA C52F EB6B 621E 9F35.) ■

openSUSE Leap 15.4

Docker, Inc. pflegt leider kein eigenes Repository für openSUSE, sondern nur für SLES. Die in openSUSE paketierte Version wäre für unsere Betrachtungen aber im Prinzip ausreichend.

Dennoch empfehlen wir die Verwendung des Zusatzrepos <http://download.opensuse.org/repositories/Virtualization:/containers/>. Zum einen könnte dort eine noch aktuellere Version zur Verfügung stehen, und zum anderen gibt es nur dort das Docker-Compose-Plugin, das wir in Abschnitt 21.6 nutzen möchten. Die Installation gestaltet sich damit wie folgt:

Schritt für Schritt: Installation von Docker CE unter openSUSE Leap 15.4

1 Einbinden des Zusatzrepos:

```
# zypper addrepo http://download.opensuse.org/repositories/\
Virtualization:containers/15.4/\
Virtualization:containers.repo
```

2 Installieren Sie nun Docker CE:

```
# zypper install -r Virtualization_containers docker ■
```

21.1.7 Installation von Podman

Das Podman-Projekt stellt selbst keine Software-Repositorys zur Verfügung. Auf allen unseren betrachteten Linux-Distributionen steht aber ein Paket namens *podman* zur Verfügung, das Sie ganz einfach über die jeweilige Paketverwaltung installieren können.

Anmerkung

Wir haben alle in dieser Einführung gezeigten Vorgehensweisen auf einem CentOS-Stream-9-System mit der Podman-Version 4.4.1 getestet.

21.1.8 Ergänzungen zur Installation, erster Systemtest

Docker: Start des Daemons und Autostart-Integration

Wenn Sie Docker einsetzen, stellen Sie nun sicher, dass der Docker-Daemon läuft und in den Autostart integriert ist. Bei Debian/Ubuntu-Systemen ist das bekanntlich nach der Installation schon gegeben; auf aktuellen SUSE- oder Red-Hat-Systemen erledigen Sie das wie gewohnt mit:

```
# systemctl enable --now docker
```

Docker: Berechtigungen

Bevor Sie mit Docker loslegen, müssen Sie noch kurz überlegen, wer in Zukunft berechtigt sein soll, Docker-Kommandos abzusetzen. Typischerweise wird hier viel als Root gearbeitet; in dem Fall müssen Sie nichts weiter tun.

Wenn Sie einem Nicht-Root-Account wie z.B. `user1` den Umgang mit Docker gestatten wollen, müssen Sie ihn in die Gruppe `docker` aufnehmen – etwa so:

```
# usermod -aG docker user1
```

Das könnte z.B. auf Ubuntu-Systemen interessant sein, um sich das ständige »`sudo ...`« zu ersparen.

Podman: Kompatibilitätsalias

Setzen Sie bitte den schon in Abschnitt 21.1.5 angesprochenen Alias (z.B. in Ihrer `~/.bashrc`):

```
alias docker=podman
```

Testaufrufe

Sie sollten nun Informationen über den laufenden Docker-Daemon erfragen können:

```
$ docker system info
```

Sie können auch das zu Testzwecken bereitgestellte »Hello World«-Image herunterladen und starten:

```
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
[...]
Hello from Docker!
This message shows that your installation appears to be working correctly.
[...]
```

Sollte dies fehlschlagen, könnte es vielleicht daran liegen, dass Sie in Ihrem Netzwerk einen Proxy benutzen müssen, um auf Ressourcen im Internet zugreifen zu können (siehe dazu gegebenenfalls Abschnitt 21.1.9).

bash-Completions

Wegen der Fülle von Möglichkeiten des `docker`-Kommandos sollten Sie auf jeden Fall passende `bash`-Completions nutzen (auf Minimalsystemen fehlen diese bisweilen schon einmal). Versuchen Sie einmal:

```
$ docker_  
[...]
```

Wenn Ihre bash jetzt mögliche Docker-Subkommandos ausgibt, ist alles in Ordnung. Ansonsten stellen Sie bitte sicher, dass das Paket *bash-completion* installiert ist, und melden sich einmal neu an. Nun sollte es gehen.

Anmerkung

Im Falle von Podman geht es leider damit noch nicht, da die Completions nicht mit einem Alias funktionieren. Podman liefert aber passende Completions mit; um diese zu nutzen, müssen Sie dann aber mit dem tatsächlichen podman-Kommando arbeiten:

```
$ source <(podman completion bash)
$ podman_ [Tab] [Tab]
[...]
```



Ergänzende Tools

Installieren Sie bitte zusätzlich schon einmal folgende Tools aus den gleichnamigen Paketen:

- ▶ *curl*
- ▶ *jq*

curl ist Kommandozeilentool, um URLs anzusprechen und Daten zu beziehen bzw. zu senden. Das ist beispielsweise beim Umgang mit REST-APIs sehr nützlich.

jq ist ein JSON-Prozessor – so eine Art Schweizer Taschenmesser für den Umgang mit Daten im JSON-Format, mit denen Sie es in der Container-Welt mitunter zu tun haben werden.

21.1.9 Betrieb hinter einem Proxy

Vor allem in Unternehmensumgebungen ist es üblich, dass der Zugriff auf das Internet nur über einen Proxy möglich ist. Die Lösung für unsere zwei Containerumgebungen ist aber recht ähnlich: Durch das Setzen geeigneter Umgebungsvariablen geben Sie der jeweiligen Software die passenden Hinweise.

Docker

Wenn Sie Docker nutzen, muss der Docker-Daemon die Umgebungsvariablen »sehen«. Auf *systemd*-basierten Distributionen lösen Sie das am besten mit einer Drop-in-Datei:

```
# mkdir -p /etc/systemd/system/docker.service.d
```

In diesem Verzeichnis erstellen Sie dann sinngemäß folgende Datei *http-proxy.conf*:

```
[Service]
Environment="HTTP_PROXY=http://proxy.example.com:3128/"
```

```
Environment="HTTPS_PROXY=http://proxy.example.com:3128/"  
Environment="NO_PROXY=localhost,127.0.0.1,hub.example.com"
```

Listing 21.1 »/etc/systemd/system/docker.service.d/http-proxy.conf«

Jetzt müssen Sie nur noch *systemd* benachrichtigen und den Docker-Daemon neu starten:

```
# systemctl daemon-reload  
# systemctl restart docker
```

Sicherstellen, dass alles geladen wurde:

```
# systemctl show --property=Environment docker
```

Damit sollte der Zugriff nun funktionieren.

Podman

Auch die Podman-Software muss mit den passenden Umgebungsvariablen versorgt werden. Eine dafür gut geeignete Stelle ist die allgemeine Konfigurationsdatei */etc/environment*, in die Sie sinngemäß folgende Zeilen eintragen:

```
HTTP_PROXY=http://proxy.example.com:3128/  
HTTPS_PROXY=http://proxy.example.com:3128/  
NO_PROXY=localhost,127.0.0.1,hub.example.com
```

Listing 21.2 »/etc/environment«

Nach einem erneuten Anmelden am System sollte der Zugriff nun funktionieren.

21.1.10 Konfiguration der Laufzeitumgebung

In diesem Abschnitt möchten wir Ihnen noch zeigen, wo und wie Sie Konfigurationseinstellungen für Ihre Container-Laufzeitumgebung vornehmen können. Wohlgermerkt: Es soll hier im Wesentlichen nicht darum gehen, *was* Sie alles einstellen können, sondern nur *wo und wie* Sie es tun können.

Docker

Bei Docker geht es hier um die Konfiguration des Docker-Daemons. Dafür stehen zwei Möglichkeiten zur Verfügung, die auch gemischt verwendet werden können (wobei Sie jedoch nicht eine identische Konfigurationsoption an beiden Orten verwenden dürfen):

- ▶ Kommandozeilenparameter des Daemons
- ▶ Konfigurationsdatei (Default: */etc/docker/daemon.json*)

Um über Kommandozeilenparameter zu parametrisieren, müssten Sie bei all unseren betrachteten Distributionen die ExecStart-Zeile der *systemd*-Service-Unit ändern. Die bevor-

zugte Variante ist also wohl eher der Weg über die Konfigurationsdatei. Als Format haben die Entwickler JSON gewählt, was technisch sicherlich gut passt, aber aus praktischer Sicht eine Fehlentscheidung war (zu viel Syntax und vor allem: keine Kommentarmöglichkeit). Einen Einstieg bzw. einen Überblick über alle Optionen finden Sie unter <https://docs.docker.com/config/daemon/>. So könnte z. B. eine reale Config aussehen, die

- ▶ das Basisverzeichnis für alle von Docker verwalteten Daten vom Default `/var/lib/docker` auf `/opt/docker` ändert und
- ▶ alle Container einen bestimmten DNS-Server verwenden lässt:

```
{
  "data-root": "/opt/docker",
  "dns": ["8.8.8.8"]
}
```

Listing 21.3 »/etc/docker/daemon.json«: Eine exemplarische Konfigurationsdatei

Bitte übernehmen Sie diese Datei aber nicht einfach so. Sie ist nur ein Beispiel, um den Aufbau einer `daemon.json` zu illustrieren. Vergessen Sie nach Änderungen an dieser Datei nicht den Restart des Dienstes:

```
# systemctl restart docker
```

Podman

Bei Podman ist die Situation weitaus unübersichtlicher, weil hier verteilte bzw. modulare Konfigurationen an den verschiedensten Stellen möglich sind. Wenn Sie selbst etwas stochern wollen: Das Verzeichnis `/etc/containers/` ist ein üblicher Startpunkt. Vorweg empfehlen wir aber dringend einen Blick in die Manpage `containers.conf(5)`.

Podman verwendet als Konfigurationsformat das nicht ganz alltägliche *TOML*, das ist so eine Art INI-Format mit der zusätzlichen Möglichkeit, verschachtelte Strukturen darzustellen. Die Projektseite <https://github.com/toml-lang/toml> bietet hier einen guten Einstieg.

21.2 Management von Images und Containern

21.2.1 Etwas Terminologie

Bevor wir zur Praxis kommen, sollten wir uns noch kurz die Zeit nehmen, einige Begriffe aus dem »Container-Universum« etwas näher zu beleuchten:

- ▶ **Image und Container**

Ein *Image* ist eine *unveränderliche* Vorlage, aus der Sie beliebig viele »lebendige« Container erzeugen können. Vielleicht ist ein Vergleich mit dem Gebiet der objektorientierten Programmierung hilfreich: »Images sind wie Klassen, und Container sind wie Objekte.«

▶ **Host**

Ein *Host* ist ganz einfach eine Maschine, auf der Container laufen.

▶ **Registry**

In der Container-Welt versteht man unter *Registry* einen Ort, an dem Images gelagert und zum Download bereitgestellt werden. Entsprechende Berechtigungen vorausgesetzt, können Images natürlich auch hochgeladen werden.

▶ **Repository**

Ein *Repository* ist eine Ansammlung von »verwandten« Images in einer Registry, typischerweise handelt es sich um verschiedene Versionen derselben Anwendung.

21.2.2 Das Command Line Interface

Das Kommandozeilentool `docker` bzw. `podman` ist bis auf Weiteres unser primäres Administrationstool. Werfen wir also einmal einen genaueren Blick darauf.

Die vielen Möglichkeiten des Kommandos werden durch Subkommandos erschlossen. Nach letzter Zählung sind es gut 50(!) Stück. Viele von ihnen haben dann noch mal Subkommandos. Die Offline-Dokumentation ist dabei jedenfalls recht gelungen. Jedes Subkommando kann mit `--help` aufgerufen werden, z.B. so:

```
$ docker system --help
$ docker system info --help
```

Und in der Regel gibt es auch jeweils noch mal Manpages mit teilweise etwas ausführlicheren Informationen:

```
$ man docker-system
$ man docker-system-info
bzw.
$ man podman-system
$ man podman-system-info
```

Um etwas mehr Ordnung hineinzubringen, wurden die Subkommandos in Docker-Version 1.13 teilweise neu strukturiert. Vor allem sind Container- und Image-spezifische Kommandos nun unterhalb von `docker container` bzw. `docker image` angesiedelt. Die »alte« Form steht in der Regel aber nach wie vor zur Verfügung:

"Alt":
\$ `docker start [...]`

"Neu":
\$ `docker container start [...]`

21.2.3 Erste Schritte: hello-world

Jedes Mal, wenn Sie das Kommando `docker run` absetzen, erzeugen Sie damit einen neuen Container:

```
$ docker run hello-world
[...]
$ docker run hello-world
[...]
```

`docker run` ist dabei nur die Kurzform für das ausführlichere `docker container run`. Aus Gründen der Bequemlichkeit wird aber meistens die alte Form verwendet.

Schauen wir also einmal nach, wie es mit unseren Containern aussieht (wir haben die Ausgabe etwas gekürzt):

```
$ docker container ls -a
CONTAINER ID   IMAGE          COMMAND         CREATED    STATUS   NAMES
43740d29e166  hello-world   [...]          Exited    naughty_northcutt
d4a488efcbd6  hello-world   [...]          Exited    mystifying_fermat
[...]
```

Die Option `-a` brauchen wir, um alle Container zu sehen (nicht nur die aktiven.) Auch hier hätten wir alternative Kommandos mit identischer Wirkung zu Verfügung: `docker container ps -a` oder das kürzere `docker ps -a`.

Das zugrunde liegende Image gibt es natürlich nur einmal:

```
$ docker image ls # 'docker images' wäre hier das alternative Kommando.
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest   f2a91732366c  4 months ago  1.85kB
```

Docker erzeugt für jeden Container einen synthetischen, eindeutigen Namen. Diese Bezeichnungen sind zwar ziemlich witzig, aber wahrscheinlich wollen Sie zu Beginn doch eher eigene, definierte Namen haben. Dies gelingt mit der Option `--name`:

```
$ docker run --name hello1 hello-world
$ docker run --name hello2 hello-world
```

Container sind im Grunde nichts »Wertvolles«: Man erzeugt sie, lässt sie laufen und wirft sie wieder weg. Deswegen hat das `run`-Kommando sogar eine Option `--rm`:

```
$ docker run --rm hello-world
```

Damit wird der Container automatisch nach Beendigung wieder gelöscht.

Untersuchen wir nun interessehalber noch, was das Kommando `docker run` unter der Haube alles tut. Es sind mehrere Schritte, und wir geben jeweils auch das Kommando an, um diese Schritte selbst einzeln zu tätigen.

1. Zunächst wird geprüft, ob das gewünschte Image lokal schon in der gewünschten Version vorhanden ist. Wenn nicht, wird es von einer externen Registry heruntergeladen:

```
$ docker image pull hello-world
```

2. Nun wird ein Container erzeugt und auf den Start vorbereitet. Unter anderem bekommt er eine eindeutige ID und einen Namen. Wenn Sie diesen nicht selbst wählen, vergibt Docker – wie schon erwähnt – synthetische Namen. Wir wählen in diesem Beispiel selbst einen Namen aus, damit wir uns im nächsten Schritt darauf beziehen können:

```
$ docker container create --name hw hello-world
```

3. Schließlich wird der Container gestartet. Um in unserem speziellen Fall die Standardausgabe sehen zu können, ist die Option `-a` nötig:

```
$ docker container start -a hw
```

21.2.4 Löschen von Containern und Images

Einen Container wird mit dem `container rm`-Subkommando gelöscht, wobei seinen Namen oder seine ID angegeben wird. Die Option `-f` ist nötig, wenn Sie aktive Container löschen möchten.

```
docker container rm [-f] <NAME_ODER_ID>
```



Tip

Wenn Sie hier mit IDs arbeiten (und keine Maus zur Hand haben), reicht es, den Anfang der ID bis auf Eindeutigkeit einzugeben.

Um alle vorhandenen Container zu löschen, lassen Sie sich zunächst mit `docker container ls -aq` die Liste der IDs generieren, und diese übergeben Sie dann direkt ans Löschkommando:

```
$ docker container ls -aq | xargs -r docker container rm -f
```

Beachten Sie im Falle von Podman bitte, dass Sie hier auf der rechten Seite der Pipe tatsächlich mit dem eigentlichen `podman`-Kommando arbeiten müssen, da `xargs` keine Aliasse aufrufen kann. (Die Option `-r` bei `xargs` bewirkt übrigens, dass das Löschkommando gar nicht gestartet wird, wenn es keinen Input – also keine Container – gibt.)

Ein Image löscht man analog mit dem `image rm`-Subkommando:

```
docker image rm [-f] <NAME>
```

Die Option `-f` braucht man hier nur, wenn es sich um Images aktiver Container handelt. Und auch dann würde das Image nicht komplett gelöscht, sondern nur seine Tags; auf diese kommen wir sogleich noch zu sprechen.

Alle vorhandenen Images zu löschen ist jetzt auch nicht mehr schwer:

```
$ docker image ls -q | xargs -r docker image rm -f
```

21.2.5 Image-Namen, Docker Hub und weitere Registrys

Image-Namen werden nicht immer so kurz und einfach bleiben wie z.B. `hello-world`. Sie können und müssen mitunter auch komplexer sein, z.B. um Eindeutigkeit zu erzielen, wenn mehrere Registrys ins Spiel kommen. Ein vollqualifizierter Image-Name (FQIN = *Fully Qualified Image Name*) sieht wie folgt aus:

```
registry_hostname[:port]/username/reponame[:tag]
```

In der Docker-Welt ist die voreingestellte Registry der Docker Hub (`docker.io`), und wenn der Username fehlt, wird hier als Default library angenommen. Die unter library angesiedelten Repositories beinhalten sogenannte *offizielle Images*, was bedeutet, dass hier sehr auf gute Dokumentation, geringen Platzbedarf, Sicherheit und Best Practices beim Bauen geachtet wird. Letztlich sollten Sie also immer erst nach offiziellen Images schauen, wenn Sie sich auf dem Docker Hub bedienen möchten. Der Zugang über `docker.io` ist übrigens nur für die API-Nutzung gedacht; als Mensch nutzen Sie besser <https://hub.docker.com/>.

Image-Tags schließlich sind *alphanumerische Bezeichner*, typischerweise sind es Versionskennungen. Immer wenn Sie kein explizites Tag angeben, wird das Tag »latest« verwendet. Beachten Sie, dass der Maintainer die Tags beliebig vergeben kann; latest muss daher nicht unbedingt die neueste Version eines Images sein. Wenn es darauf ankommt, sollten Sie das gewünschte Tag sowieso immer explizit angeben. Zusammenfassend sind in der Docker-Welt beispielsweise die folgenden beiden Aufrufe äquivalent:

```
$ docker pull httpd
$ docker pull docker.io/library/httpd:latest
```

Registrierung auf dem Docker Hub

Docker bietet verschiedene Arten der Mitgliedschaft an, beginnend mit einer kostenlosen Registrierung, für die Sie nur eine E-Mail-Adresse benötigen. Damit können Sie dann bereits öffentliche Repositories anlegen, und Sie haben ein etwas höheres Rate-Limit für Image Downloads (200/6h) als beim anonymen Zugriff. Die Vorteile der kostenpflichtigen Mitgliedschaften entnehmen Sie bitte der Seite <https://www.docker.com/pricing>.

Weitere öffentlich verfügbare Registrys

Es gibt einige weitere Anbieter, die Images hosten. Meistens werden diese Angebote bei der Arbeit mit Cloud-Diensten genutzt. Bekannt sind beispielsweise:

- ▶ Quay (<https://quay.io/>)

- ▶ Google Container Registry (<https://cloud.google.com/container-registry/>)
- ▶ AWS Container Registry (<https://aws.amazon.com/ecr/>)

Auf den Betrieb eigener, privater Registrys kommen wir in Abschnitt 21.7 noch zu sprechen.

21.2.6 Handling von Containern

Interaktive Container

Wir wollen nun den Umgang mit interaktiven Containern demonstrieren. Als Beispiel verwenden wir ein Alpine Linux, das schon außerhalb der Containerwelt für seine Schlankeheit bekannt ist:

```
$ docker run -it --name a1 alpine
```

Bei diesem Aufruf sorgt `-i` dafür, dass eine interaktive Bedienung möglich ist, also dass STDIN offengehalten wird. Mit `-t` wird ein Pseudo-Terminal alloziert. Sie befinden sich nun »im Container«:

```
/ # ls
bin  etc  lib  mnt  proc  run  srv  tmp  var
dev  home media opt  root  sbin sys  usr
/ # ps -ef
PID  USER  TIME  COMMAND
   1  root   0:00  /bin/sh
   8  root   0:00  ps -ef
/ # du -sh /
5.5M  /
/ # exit
```

Durch das `exit`-Kommando haben wir den Hauptprozess des Containers beendet – und damit auch den Container selbst. Einen erneuten interaktiven Start *desselben* Containers erreichen Sie mit:

```
$ docker start -ai a1
```

`-a` führt dazu, dass STDOUT/STDERR wieder verbunden wird, `-i` verbindet STDIN erneut. Hätten Sie keinen eigenen Namen wie »a1« vergeben, müssten Sie den synthetischen Namen oder die Container-ID verwenden. Im letzteren Fall wäre auch eine Abkürzung möglich, die aber eindeutig sein muss.

Detach

Mit der Tastensequenz `[Strg]+[P]` `[Strg]+[Q]` können Sie sich von einem interaktiven Container lösen, ohne ihn beenden zu müssen. Danach gehen alle Eingaben wieder ans Hostsystem, aber unser Container läuft weiter. Falls wir erneut interaktiv mit ihm arbeiten wollen, geben wir Folgendes ein:

```
$ docker attach a1
/ #
```

Anmerkung

Bei manchen interaktiven Linux-Containern kann es schon mal sein, dass Sie nach dem attach noch eine Taste (z. B. `Enter`) drücken müssen, damit die Eingabeaufforderung sichtbar wird.



21

Befehle »von außen« mit »docker exec«

Es gibt auch die Möglichkeit, mit `docker [container] exec` in einem *laufenden* Container quasi direkt »von außen« Befehle aufzurufen:

- ▶ Simpler Kommandoaufruf:


```
$ docker exec a1 echo "Hallo aus dem Container"
```
- ▶ Wenn Shell-Mechanismen involviert sind, müssen Sie das Ganze über eine Shell aufrufen:


```
$ docker exec a1 sh -c "ls -l > ls.out"
```
- ▶ Wenn der Container per STDIN Daten von außen empfangen soll, brauchen Sie die `-i`-Option:


```
$ seq 10 | docker exec -i a1 sort -rn
```
- ▶ Und wenn die zu startende Anwendung ein richtiges Terminal benötigt, brauchen Sie zusätzlich noch die `-t`-Option:


```
$ docker exec -it a1 vi test.txt
```

Interaktive Shell im Container

Sie werden `docker exec` wohl am häufigsten nutzen, um eine interaktive Shell in einem Container aufzurufen:

```
$ docker exec -it <CONTAINER> sh
```

Dies gibt Ihnen eine Shell, die Sie auf ganz gewöhnlichem Wege wieder verlassen können. Und wenn im Container andere Shells (z. B. `bash`) vorhanden sind, können Sie selbstverständlich auch diese starten.

Service-Container

Die meiste Zeit werden Sie wahrscheinlich Container betreiben, die einen Dienst anbieten (wie beispielsweise Webserver oder Datenbankserver). Diese sind in erster Linie nicht zum interaktiven Arbeiten gedacht und werden deswegen mit der Option `-d` zumeist im *Detached Mode* (also im Hintergrund) gestartet. Hier sehen Sie den Start am Beispiel eines Apache-Containers:

```
$ docker run -d --name httpd httpd
```

Ein Start mit dem bereits bekannten Optionspaar `-it` wäre hier relativ nutzlos, da der Hauptprozess in diesen Containern keine Shell ist.



Anmerkung

Im Falle von Podman bekommen Sie hier ggf. ein Auswahlmenü, wenn ein Image namens `httpd` in mehreren Registries gefunden wird. Wählen Sie dann `docker.io/library/httpd:latest`.

21.2.7 Prozessverwaltung

Wir starten exemplarisch ein Alpine Linux und darin ein paar zusätzliche Prozesse:

```
$ docker run -itd --name a1 alpine
```

```
$ docker container exec -d a1 sleep 1000
```

```
$ docker container exec -d a1 sleep 2000
```

`docker [container] top` zeigt Ihnen alle Prozesse eines Containers:

```
$ docker container top a1
```

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	21108	21092	0	11:39	pts/0	00:00:00	/bin/sh
root	21149	21092	0	11:40	?	00:00:00	sleep 1000
root	21171	21092	0	11:40	?	00:00:00	sleep 2000

Die hier gezeigte Ausgabe kommt von einem Docker-System. Wenn Sie Podman einsetzen, würden Sie hier Prozess-IDs aus der internen Sicht des Containers sehen, also auf jeden Fall eine PID 1 und ein paar weitere »kleine« PIDs. (Dies wird intern durch die Verwendung von Linux-Kernel-Namespaces ermöglicht.)

Um diese Sicht auch mit Docker zu bekommen, könnten Sie ein `ps`-Kommando *im Container* starten:

```
$ docker container exec a1 ps -ef
```

PID	USER	TIME	COMMAND
1	root	0:00	/bin/sh
13	root	0:00	sleep 1000
17	root	0:00	sleep 2000
37	root	0:00	ps -ef

Das ist aber leider nicht bei jedem Container möglich. Scheitern würde es in der Praxis oft daran, dass gar kein `ps`-Kommando vorhanden ist. (»Das ist doch ein Standard-Tool?« Ja – aber nicht in einem Container!)

Unter den gängigen Linux-Images sind es hauptsächlich Debian- bzw. Ubuntu-Varianten, die standardmäßig ohne Prozess-Tools angeboten werden. Falls Sie Ihren Container dann doch einmal im Nachhinein damit ausstatten wollen: Das entsprechende Paket heißt hier *procps*.

Docker: Monitoring mit »ctop«

Auf GitHub gibt es das äußerst gelungene Monitoring-Tool *ctop*, das aus technischen Gründen leider nur im Zusammenspiel mit Docker funktioniert.

Schauen Sie doch mal bei <https://github.com/bcicen/ctop> vorbei, und laden Sie sich ein fertiges Executable herunter (siehe den dortigen Abschnitt »Install«). Etwa so:

```
# VERSION="0.7.7"
# wget https://github.com/bcicen/ctop/releases/download/v$VERSION/ctop-$VERSION-linux-amd64 -O /usr/local/bin/ctop
# chmod 755 /usr/local/bin/ctop
```

Nützliche *ctop*-Optionen sehen Sie in Tabelle 21.1:

Option	Bedeutung
-a	Nur aktive Container anzeigen
-i	Farbschema invertieren
-f <STRING>	angezeigte Container filtern

Tabelle 21.1 »ctop«-Optionen

21.2.8 Umgebungsvariablen

Mitunter gibt es die Anforderung, einen Container beim Start von außen mit irgendwelchen Informationen zu versorgen. Beispielsweise benötigt ein Datenbank-Container möglicherweise ein Master-Passwort, um sich beim ersten Start initialisieren zu können.

Eine oft angewandte Vorgehensweise ist die Übergabe von *Umgebungsvariablen*. Es gibt dafür drei Möglichkeiten:

- ▶ Übergabe von einzelnen Variablen mit Wert:


```
docker run -e <VAR1=WERT1> [-e <VAR2=WERT2> ...] [...]
```
- ▶ Übergabe einer Datei, die viele »VAR=WERT«-Zeilen enthalten kann:


```
docker run --env-file <FILENAME> [...]
```
- ▶ Weitergabe einer Umgebungsvariablen aus dem Hostsystem:


```
docker run -e <VAR> [...]
```

21.2.9 Logging

In Containern gilt folgender Standard für die Verarbeitung von Logging-Infos: Alle Ausgaben, die Prozess Nr. 1 nach STDOUT oder STDERR schreibt, werden von einem Logging-Treiber »eingefangen« und entsprechend der treiberspezifischen Konfiguration weiterverarbeitet. Sie können die Logs eines Containers jederzeit interaktiv betrachten mit:

```
docker [container] logs <CONTAINER>
```

Hierbei stehen Ihnen auch einige nützliche Filter- bzw. Anzeigeeoptionen zur Verfügung, die Sie in Tabelle 21.2 sehen:

Option	Bedeutung
-f	Log-Output verfolgen
-n <ZAHL>	Anzahl Zeilen vom Ende des Logs anzeigen
--since <ZEITPUNKT>	Logzeilen seit diesem Zeitpunkt anzeigen
--until <ZEITPUNKT>	Logzeilen bis zu diesem Zeitpunkt anzeigen

Tabelle 21.2 Filter- und Anzeigeeoptionen von »docker logs«

Die Hilfe zu `docker logs` gibt Ihnen Informationen zum Format der Zeitpunktspezifikation. Erfreulicherweise funktionieren aber auch relative Angaben wie 4h oder 15m!

Ob der Container eine Logmeldung nach STDOUT oder STDERR geschrieben hat, wird von `docker logs` übrigens widerspiegelt:

Nur Fehler zeigen:

```
docker logs <CONTAINER> 1>/dev/null
```

Nur normale Meldungen zeigen:

```
docker logs <CONTAINER> 2>/dev/null
```

Logging-Treiber

- ▶ Unter Docker ist `json-file` der voreingestellte Logging-Treiber, d. h., die Meldungen landen in einer (rotierten) JSON-Datei. Welche das ist, sagt Ihnen:

```
docker inspect --format='{{.LogPath}}' <CONTAINER>
```

- ▶ Unter Podman werden Container-Logs standardmäßig an Journald weitergegeben (Treiber `journald`).

Der gewünschte Treiber ist über den Startparameter `--log-driver` einstellbar; mit dem Parameter `--log-opt=[]` können Sie treiberspezifische Einstellungen ändern. Natürlich können Sie auch in der globalen Runtime-Konfiguration Vorgaben für alle Container machen.

Eine wirklich große Zahl von Logging-Treibern bietet zurzeit nur Docker; bei Interesse finden Sie weitere Infos unter <https://docs.docker.com/config/containers/logging/configure/#supported-logging-drivers>.

21.2.10 Verteilung von Images über Dateiversand

Wollen Sie ein Image einfach und ohne Informationsverlust von einem Host auf einen anderen bringen, so können Sie mit `docker [image] save` bzw. `load` arbeiten. Um Platz zu sparen, empfiehlt sich in der Regel eine On-the-fly-Kompression:

```
Host 1:
$ docker image save <IMAGE> | gzip > <name>.tar.gz
```

[... dann Dateitransfer nach Host 2 ...]

```
Host 2:
$ docker image load -i <name>.tar.gz
```

Sofern Host 2 »kompatibel« zu Host 1 ist, können Sie aus dem Image jetzt Container erzeugen und starten. Wenn der zweite Docker Host via SSH erreichbar ist, geht das alles noch viel eleganter:

```
Host 1:
$ docker image save <IMAGE> | ssh <HOST_2> docker image load
```

21.2.11 Ausgaben filtern und/oder formatieren

Sehr viele Docker-Subkommandos bieten die Möglichkeit, die Ausgabe zu filtern bzw. zu formatieren. Die Online-Hilfe liest sich dann typischerweise etwa so:

```
Options:
[...]
  -f, --filter filter  Filter output based on conditions provided
  --format string     Pretty-print containers using a Go template
[...]
```

Filtern

Mit einem Ausgabefilter können Sie die Liste der Ergebnisobjekte reduzieren. Die konkreten Filtermöglichkeiten sind dabei vom jeweiligen Docker-Subkommando abhängig; sehen Sie z.B. hier alle Möglichkeiten für `docker container ls` bzw. `docker image ls`:

- ▶ <https://docs.docker.com/engine/reference/commandline/ps/#filtering>
- ▶ <https://docs.docker.com/engine/reference/commandline/images/#filtering>

Beispielsweise schweigt sich die Doku an diesen Stellen aber darüber aus, dass der `name`-Filter ein regulärer Ausdruck sein kann:

Zeige alle Container, die ein "a" im Namen haben:

```
docker container ls -f name="a"
```

..., deren Name mit "a" anfängt:

```
docker container ls -f name="^a"
```

Etwas knifflig ist auch die Logik, wenn Sie `-f` wiederholt anwenden: Gleiche Filter werden zunächst ODER-verknüpft, verschiedene werden danach UND-verknüpft.

Formatieren mit Go-Templates

Über Go-Templates (<https://golang.org/pkg/text/template/>) eröffnen sich recht komplexe Möglichkeiten, die Ausgabe nach eigenen Wünschen zu gestalten. Denn wie bei allen Template-Engines verbirgt sich dahinter fast eine komplette Programmiersprache. Wir geben hier nur ein paar Aufrufbeispiele, um die Möglichkeiten zu demonstrieren (bitte selbst ausprobieren):

```
$ docker image inspect httpd --format '{{.Config.Cmd}}'  
[httpd-foreground]
```

```
$ docker image inspect httpd --format '{{.Config.ExposedPorts}}'  
map[80/tcp:{}]
```

```
$ docker image inspect httpd --format '{{json .Config.Env}}' | jq  
[  
  "HTTPD_PREFIX=/usr/local/apache2",  
  "NGHTTP2_VERSION=1.18.1-1",  
  "OPENSSL_VERSION=1.0.2l-1~bpo8+1",  
  "HTTPD_VERSION=2.4.33",  
  [...]  
]
```

Alle IP-Adressen des Containers "h1":

```
$ docker container inspect h1 \  
  --format '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'  
[...]
```

"container ls" mal anders (bei vielen Containern unübersichtlich):

```
$ docker container ls -a --format '{{json .}}' | jq  
[...]
```

Bestimmen, welche Spalten in die Ausgabe kommen:

```
$ docker container ls -a --format '{{printf "%-20s%-20s" .Names .RunningFor}}'
```



```
[...]
Oder noch einfacher so:
$ docker container ls -a --format 'table {{.Names}}\t{{.RunningFor}}'
[...]
```

21.2.12 Restart-Policys: Verhalten beim Host-Restart

Eine Frage, die sich noch stellt, ist die folgende: Wie ist der Status von (vormals aktiven) Containern nach einem Systemneustart?

Schnelle Antwort: Sie sind alle beendet und werden auch nicht von alleine wieder gestartet. Wenn man sich auf den simplen Standpunkt zurückzieht, dass Container letztlich nur Prozesse sind, war das auch zu erwarten.

Nun wäre ein automatisches »Wiedererwachen« aber sicher oft wünschenswert. Docker hat dazu die sogenannten *Restart-Policys* eingeführt; eine solche können Sie beim Start eines Containers (oder auch später mit `docker [container] update`) festlegen:

```
$ docker run --restart <POLICY> [...]
```

Es stehen folgende Policys zur Verfügung:

- ▶ **no**
Der Default: Keine automatischen Restarts.
- ▶ **always**
Der Container wird auf jeden Fall wieder gestartet.
- ▶ **unless-stopped**
Der Container wird wieder gestartet, außer er ist zuvor explizit gestoppt worden.
- ▶ **on-failure[:max-retries]**
Der Container wird wieder gestartet, wenn er sich mit einem Fehler beendet hat (Exitcode ungleich 0).

Docker: Live Restore

Docker bringt zudem noch ein interessantes Feature namens *Live Restore* mit, wodurch laufende Container einen Docker-Service-Restart »überleben« können. Das ist hauptsächlich gedacht, um Downtimes von Containern bei Patch-Release-Updates zu vermeiden. Das Ganze wird in der Service-Konfiguration aktiviert und ist wirklich sehr empfehlenswert:

```
{
  [...],
  "live-restore": true
}
```

Listing 21.4 »/etc/docker/daemon.json«: Aktivieren des Live-Restore-Features

Und Podman?

Wenn Sie Podman nutzen, haben Sie natürlich das Problem, dass es keine zentrale Instanz wie den Docker-Daemon gibt, die den Status von Containern überwachen kann. Aber eigentlich müsste man bei bzw. nach einem Systemneustart nur herausfinden, welche Container entsprechende Restart-Policies haben, und diese dann einfach wieder starten, z. B. so:

```
$ podman container ls -a -f restart-policy=always
```

```
$ podman container start --all -f restart-policy=always
```

Nichts davon würde übrigens mit Docker funktionieren, aber muss es ja auch nicht. Möglicherweise müssen Sie nun auch gar nicht selbst eine Service-Unit schreiben; unter CentOS Stream 9 jedenfalls wird schon eine mitgeliefert, die Sie bei Bedarf nur aktivieren müssten:

```
# systemctl enable podman-restart
```



Anmerkung

Seit der Podman-Version 4.4 gibt es noch eine schwergewichtigere Lösung: Sie können sich mit `podman generate systemd` eigene Service-Units für Container generieren lassen und diese dann manuell in `systemd` integrieren. Das kann ein Weg für sehr überschaubare Umgebungen sein, wenn maximale Flexibilität gefordert ist.

21.2.13 Container limitieren

Standardmäßig hat ein Container – wie jeder andere Prozess auch – unbegrenzten Zugang zu allen Host-Ressourcen wie Hauptspeicher, CPU oder Block-IO. Mit `docker [container] stats` können Sie sich zunächst einmal einen kompakten Überblick über den Ressourcenverbrauch verschaffen:

```
$ docker container stats -a --no-stream
CONTAINER ID  NAME  CPU %  MEM USAGE/LIMIT  MEM %  NET I/O  [...]
46f3276f7591  h1    0.00%  10.66MiB/1.788GiB  0.58%  428kB/628kB  [...]
[...]
```

Wenn Sie `ctop` installiert haben, werden Sie diesem wahrscheinlich aber den Vorzug geben (siehe Abschnitt 21.2.7). Um nun zu vermeiden, dass einzelne Container »über die Stränge schlagen«, ist es möglich, deren Ressourcen zu limitieren. Die dafür beim Containerstart vorgesehenen Optionen können Sie im Großen und Ganzen wie folgt auf den Bildschirm bringen:

```
$ docker run --help | egrep -i 'limit|cpu|memory|blkio|device'
--cpuset-cpus string          CPUs in which to allow execution
-m, --memory bytes           Memory limit
```

```
[...]
... jede Menge anderer Optionen ...
[...]
```

Damit eine Ressource limitiert werden kann, muss der zugrunde liegende Host-Kernel dies aber auch unterstützen. Oft fehlt das Feature *Swap limit support*; prüfen Sie dies auf Ihrem System:

```
$ docker system info
[...]
WARNING: No swap limit support
```

Bei Bedarf können Sie dieses Feature mit den zwei Kernelparametern

```
cgroup_enable=memory swapaccount=1
```

aktivieren. Dies geschieht wie gewohnt in der Datei */etc/default/grub*, und zwar mit der Variablen `GRUB_CMDLINE_LINUX_DEFAULT`. Vergessen Sie danach nicht, die GRUB-Config neu zu generieren. Oben in der Datei finden Sie einen Kommentar, der Ihnen zeigt, wie das auf Ihrem Linux-System zu erledigen ist. Um die Limitierungen eines bereits laufenden Containers zu ändern, steht `docker [container] update` zur Verfügung. Sie können nicht alle Starteinstellungen ändern, aber doch die wesentlichsten. In jedem Fall sollten Sie auch die Online-Dokumentation unter https://docs.docker.com/config/containers/resource_constraints/ lesen, wenn Sie sich genauer mit dem Thema beschäftigen möchten!

Ein Beispiel zum Nachspielen

Wir gehen exemplarisch einmal davon aus, dass Ihr Docker Host 4 CPU-Kerne zur Verfügung hat. Wenn Sie das in einer virtuellen Testumgebung exakt so nachspielen möchten, dann empfiehlt sich nach einer eventuellen Änderung der Kern-Anzahl ein Reboot!

Schritt für Schritt: Beispiel zum Limitieren von Containern



1 *Starten Sie einen Debian-Container:*

```
$ docker run -itd --name deb1 debian:stable-slim
```

2 *Statten Sie ihn mit ein paar Tools aus:*

```
$ docker exec deb1 apt-get update
$ docker exec deb1 apt-get -y install stress-ng
```

3 *Container beobachten:*

Lassen Sie in einem zweiten Fenster ein `ctop -f deb1` laufen, falls vorhanden. Ansonsten verwenden Sie `docker stats deb1`.

4 Erzeugen Sie im Container etwas CPU-Last:

```
docker exec -it deb1 stress-ng -c 0
```

Das wird die zur Verfügung stehenden Kerne jeweils zu 100 % auslasten.

5 Probieren Sie in einem dritten Fenster einige CPU-Limitierungen aus:

Einschränkung auf die ersten beiden CPUs:

```
$ docker update --cpuset-cpus 0,1 deb1
```

Rückgängig:

```
$ docker update --cpuset-cpus 0-3 deb1
```

Limitierung der Rechenleistung auf 2 1/2 CPUs:

```
$ docker update --cpus 2.5 deb1
```

Rückgängig:

```
$ docker update --cpus 4 deb1
```

6 Brechen Sie »stress-ng« ab, und erzeugen Sie nun Memory-Stress:

```
docker exec -it deb1 stress-ng --vm 1 --vm-bytes 256m
```

7 Versuchen Sie Memory-Limits im Wechsel mit diversen »Stress-Situationen«:

Höchstens 512 MB; aber unlimitierter Swap erlaubt:

```
$ docker update --memory 512m --memory-swap -1 deb1
```

Höchstens 512 MB; kein Swap erlaubt:

```
$ docker update --memory 512m --memory-swap 512m deb1
```

Höchstens 512 MB; zusätzlich 256 MB Swap erlaubt:

```
$ docker update --memory 512m --memory-swap 768m deb1
```

**Achtung**

Wenn Ihr System keinen `swap limit` support hat, dann wird der Swap *überhaupt nicht* limitiert. Das ist wahrscheinlich nicht das, was Sie wollen!

**Anmerkung**

Podman konnte in der getesteten Version mit Memory-Limit-Updates nicht umgehen. Aber das ist ohnehin akademisch – Sie würden solche Limits sowieso stets beim Containerstart spezifizieren.



Wie Sie sich sicher vorstellen können, ist es hochproblematisch, einer »normalen« Anwendung im laufenden Betrieb Speicher wegzunehmen; das würde wohl meistens einen Absturz

nach sich ziehen. Davon abgesehen sollten Sie bereits in der Testphase genau evaluieren, wie sich Ihre Anwendung verhält, wenn ein Speicherlimit erreicht wird bzw. überschritten werden müsste!

Docker: Plattenplatz limitieren

Unter gewissen Umständen lässt sich unter Docker auch der für einen Container zur Verfügung stehende Plattenplatz limitieren. Wenn Sie einen `overlay2`-Storage-Treiber einsetzen (siehe `docker system info`), dann muss Ihr Backing-Filesystem `xfs` mit der Mount-Option `pquota` sein, damit es funktioniert:

```
$ docker run [...] --storage-opt size=10G [...]
```

Beim `devicemapper`-, `btrfs`- und `zfs`-Treiber geht es auch, allerdings kann man dort keinen Wert konfigurieren, der kleiner ist als die `Default-BaseFS-Size`.

21.2.14 Packungsdichte

Ein oft gepriesenes Feature der Containertechnologie ist die erhöhte Packungsdichte im Vergleich zu klassischen VMs. Natürlich können Sie auf einem Host, der vielleicht 10 VMs verkraften würde, möglicherweise Hunderte von Containern unterbringen. Das gilt aber zunächst nur in Bezug auf den Platzbedarf. Die anderen zur Verfügung stehenden Ressourcen wie CPU, RAM und Netzwerk vervielfachen sich meist leider nicht automatisch. Aber dennoch, nur um Leute zu beeindrucken, könnte man auf einem noch so minimal ausgestatteten Host »mal eben« 25 Apache-Container starten:

```
$ for i in $(seq 0 24)
do
  docker run -d --name httpd$i -p $((8080+i)):80 httpd \
  bash -c "echo '<h1>Hello from httpd$i</h1>' \
    >/usr/local/apache2/htdocs/index.html; \
    exec httpd-foreground"
done
```

Die Services sind dann über den Host auf den Ports 8080, 8081, ..., 8104 erreichbar.

21.2.15 Systeminformationen und Aufräumarbeiten

Den Befehl `docker system info` haben wir ja schon gesehen – mit ihm lassen sich Informationen zum System abrufen. Mit `docker system df` erhalten Sie eine Übersicht über den Platzbedarf von Images, Containern und Volumes (zu Letzteren kommen wir in Kürze). Mit der Option `-v` werden diese Informationen noch detaillierter dargestellt. `docker system prune` schließlich dient dazu, nicht mehr benutzte Docker-Objekte zu entfernen (und damit wieder

Plattenplatz freizugeben). Mit der Option `--volumes` werden auch Volumes mit einbezogen. Dieses Kommando ist mit absoluter Vorsicht zu genießen!

21.3 Docker-Networking

Wenn Sie auf einem Host mit mindestens einem aktiven Container einmal die Netzwerk-Interfaces betrachten, sehen Sie zusätzlich zu den »normalen« Interfaces wie `lo`, `eth*`, `ens*`, ... Folgendes:

- ▶ ein Interface namens `docker0` bzw. `podman0`
- ▶ ein oder mehrere Interfaces namens `veth*`

Ersteres ist ein Bridge-Device, über das die Container mit der Außenwelt kommunizieren können. Per Default bekommt es unter Docker eine private Class-B-Adresse zugewiesen (in der Regel `172.17.0.1`, wenn netzwerktechnisch nichts dagegen spricht). Unter Podman ist es eine Class-A-Adresse (`10.x.0.1`). Mit dem `ip`-Kommando können Sie das alles genauer überprüfen (hier am Beispiel von Docker):

```
# ip -br -c link show type bridge
docker0          UP    02:42:a4:7c:02:d8 <BROADCAST,MULTICAST,UP,LOWER_UP>

# ip -br -c link show master docker0
veth09301da@if114 UP    0e:05:0f:70:68:a3 <BROADCAST,MULTICAST,UP,LOWER_UP>
veth8162249@if118 UP    fa:7e:ee:a9:5c:32 <BROADCAST,MULTICAST,UP,LOWER_UP>
```

Die `veth*`-Interfaces sind jedem Container individuell zugeordnet und entsprechen der jeweiligen `eth0`-Schnittstelle auf der »Innenseite«. Abbildung 21.1 verdeutlicht die Situation.

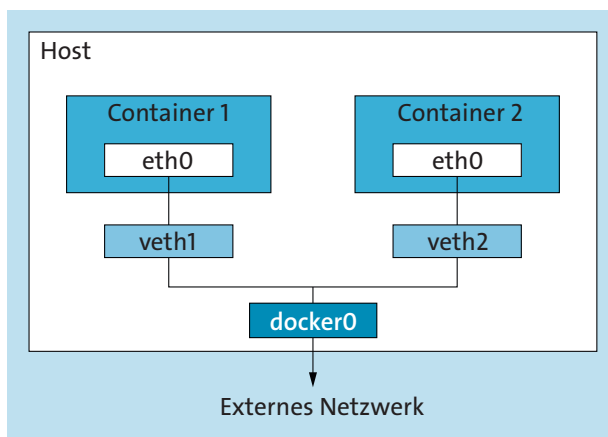


Abbildung 21.1 »veth*«-Interfaces in Docker

Jetzt fragt sich nur noch, ob und wie die Container externe Netzwerke wie das Internet erreichen können, denn die private IP-Adresse des Bridge-Interfaces taugt ja nicht zur Kommunikation. Dies gelingt mithilfe von NAT/Masquerading-Regeln im Linux-Paketfilter. Wenn Sie sich auskennen und etwas stochern wollen, dann rufen Sie `nft list table nat` auf und schauen in die `POSTROUTING`-Chain.

Achtung

Das alles ist besonders dann beachtenswert, wenn Sie auf Ihrem Host die Firewall manipulieren (z. B. neu starten) – das kann die Container-Kommunikation durchaus stören. Im Falle von Docker hilft es oft, danach den Docker-Service neu zu starten.



21.3.1 User Defined Networks

Was bislang noch nicht klar ist: Wie sollen Container nun wirklich *miteinander* kommunizieren? Die IP-Adressen sind ja im Prinzip zufällig, und kein Container kennt »einfach so« die IP-Adresse seiner Container-Kollegen.

Die Lösung sind hier die sogenannten *User Defined Networks*. Neben dem Default-Bridge-Network können Sie beliebig weitere anlegen, und alle Container, die mit demselben Netzwerk verbunden sind, können sich »sehen« – und sich über einen impliziten DNS-Server sogar mit Namen erreichen!

Mit dem Kommando `docker network` verwalten Sie Ihre User Defined Networks. Es hat relativ logische Subkommandos:

Usage: `docker network COMMAND`

Commands:

```
connect    Connect a container to a network
create     Create a network
disconnect Disconnect a container from a network
inspect    Display detailed information on one or more networks
ls         List networks
prune      Remove all unused networks
rm         Remove one or more networks
```

Beginnen wir erst einmal mit einer Auflistung:

```
$ docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
daeae82b9f4f	bridge	bridge	local
994da79d4b61	host	host	local
ad0818e3920e	none	null	local

Hinter den verschiedenen Netzwerken arbeiten verschiedene Treiber, wobei der `bridge`-Treiber für uns momentan der nützlichste ist. Docker-spezifisch ist der `host`-Treiber, mit dem Sie eine direkte Verbindung mit dem Host-Netzwerk schaffen könnten. Es gäbe dabei also netzwerktechnisch keine Isolierung, was nur in speziellen Situationen ratsam ist. Der ebenfalls Docker-spezifische `null`-Treiber bietet gar kein Netz und kommt demzufolge ebenfalls sehr selten zur Anwendung.

Legen wir also mal ein neues Netzwerk an (der Default-Treibertyp ist `bridge`):

```
$ docker network create mynet
```

Und gleich noch mal zwei Alpines, beide mit `mynet` verbunden:

```
$ docker run -itd --name a1 --rm --network mynet alpine
$ docker run -itd --name a2 --rm --network mynet alpine
```

Wenn Sie sich nun mit einem der beiden verbinden, könnten Sie den anderen mit seinem Namen ansprechen (z.B. `pingen`).

21.3.2 Portmapping

Eine weitere, noch offene Frage lautet: Wenn Sie einen Netzwerkdienst wie Apache in einem Container betreiben, wie können Sie ihn dann »von außen« erreichen? (Und zwar *ohne* ihn – was im Falle von Docker möglich wäre – an das Host-Netzwerk zu binden. Das würde zwar gehen, steht aber hier nicht zur Diskussion.)

Eine gangbare Lösung heißt *Portmapping*. Das bedeutet, die intern offenen Ports des Containers mit Ports des Hosts zu verbinden, die dann von außen erreichbar sind (sofern nichts anderes noch dagegen spricht, wie etwa eine Firewall).

Betrachten wir das am Beispiel von `httpd`. So verbinden Sie exemplarisch den internen Port 80 mit dem Port 8000 auf Ihrem Host:

```
$ docker run -d --name httpd -p 8000:80 httpd
```

Woher können Sie aber in dieser Situation wissen, dass der Container intern den Port 80 öffnet?

- ▶ Sie vertrauen auf die Ausgabe von

```
$ docker image inspect httpd -f '{{.Config.ExposedPorts}}'
map[80/tcp:{}]
```
- ▶ (Besser:) Sie lesen die Dokumentation des Images.

Falls das Image mehrere Ports freigibt und Sie alle verbinden möchten, wiederholen Sie einfach die Option `-p`. Es gibt auch noch den Schalter `-P`, der einfach freie hohe Ports auf dem Host auswählen würde. Das ist aber für unsere Situation eher uninteressant.

Erweiterte Möglichkeiten beim Portmapping

Tabelle 21.3 zeigt zwei Beispiele für weitere Möglichkeiten beim Portmapping.

Beispiel	Bedeutung
-p 127.0.0.1:8000:80	Port 8000 nur an die Host-IP 127.0.0.1 binden
-p 5300:53/udp	UDP-Port-Mapping

Tabelle 21.3 Weitere Möglichkeiten beim Portmapping

Reihenfolge

Bei der Angabe des Portmappings sollten Sie sich natürlich irgendwie merken, in welcher Reihenfolge die Ports angegeben werden müssen. Wie wir gesehen haben, ist das:

```
<HOST_PORT>:<CONTAINER_PORT>
```

Auch später bei Volume-Mounts bleibt die Reihenfolge konsistent identisch: **Erst der Host, dann der Container**. Sie könnten sich also eine Eselsbrücke bauen wie:

Der Host ist wichtiger, und das Wichtige kommt immer zuerst!

21.3.3 »/etc/hosts«-Einträge beim Containerstart

Ein Feature »am Rande«, das auch mal nützlich sein kann: Wenn Sie Containern lokale Namensauflösungen mit auf den Weg geben möchten (d. h. */etc/hosts*-Einträge im Container), so steht Ihnen dazu die Option `--add-host` zur Verfügung:

```
$ docker run -it --name a3 --rm \
  --add-host "dns1.example.org:8.8.8.8" alpine
```

Seit Docker-Version 20.10 gibt es auch das spezielle Kürzel `host-gateway`, das anstatt einer IP-Adresse angegeben werden kann. Docker setzt dann an diese Stelle automatisch die richtige IP-Adresse des Hosts aus Sicht des Containers. Das ist extrem nützlich, falls dieser Container einmal mit dem Host interagieren will:

```
$ docker run -it --name a3 --rm \
  --add-host "host.containers.internal:host-gateway" alpine
```

Der Name aus Sicht des Containers ist natürlich beliebig wählbar. Wir haben den etwas sperrigen Namen `host.containers.internal` deswegen gewählt, weil dieser Name unter Podman bereits *automatisch* in jeder Container-*/etc/hosts* enthalten ist.

21.4 Containerdaten und Persistenz

In diesem Abschnitt wollen wir uns der Datenhaltung in Containern zuwenden. Für ein besseres Verständnis ist aber zunächst ein kurzer Blick auf die Innereien von Images und Containern hilfreich.

21.4.1 Aufbau von Images und Containern

Ein Image besteht im Wesentlichen aus einem Stapel von Verzeichnis-Schichten zuzüglich einiger Meta-Informationen. Die Schichten des Images sind alle unveränderlich (read-only).

Abbildung 21.2 zeigt ein hypothetisches Image einer ebenso hypothetischen Webanwendung.

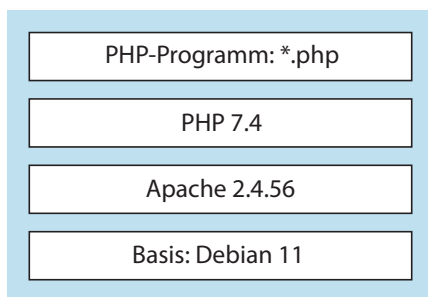


Abbildung 21.2 Schichten eines Images

Wenn Sie aus einem Image nun einen Container erzeugen, so wird *on top* eine weitere Schicht hinzugefügt, die im Unterschied zu den Image-Schichten auch schreibbar ist und in der somit neue Daten abgelegt werden können. Diese Schicht macht im Prinzip den Unterschied zwischen einem Image und einem Container aus, jedenfalls soweit es nur um die Daten geht.

Und wie funktioniert nun der Zugriff auf Dateien in diesem »Schichtsalat«? Im Prinzip ganz einfach: Die Schichten werden der Reihe nach von oben nach unten durchsucht; der erste Treffer gewinnt. Ganz neue Dateien werden einfach in der obersten Read-Write-Schicht angelegt, und um auch bereits existierende Dateien aus den unteren Read-only-Schichten »ändern« zu können, wird das *Copy-On-Write*-Verfahren angewendet: In der obersten Read-Write-Schicht wird eine Kopie angelegt, und diese kann dann verändert werden.

Wird ein Container schließlich wieder gelöscht, so wird auch seine Read-Write-Schicht freigegeben, und alle neuen bzw. geänderten Daten sind unwiederbringlich verloren. Das ist natürlich ein gewisses Problem, dessen Lösung uns in den folgenden Unterabschnitten beschäftigen wird.

Vorteile dieses Konzepts

Um einen neuen Container zu erstellen, muss keine komplette Kopie eines Images erzeugt werden, sondern es reicht, eine Liste von Referenzen auf die entsprechenden Image-Schichten zu erstellen – zuzüglich einer anfänglich leeren Read-Write-Schicht.

- ▶ Das bedeutet eine immense Ersparnis von Plattenplatz.
- ▶ Das ermöglicht ein extrem zügiges Startverhalten von Containern.

Mit gewisser Berechtigung lässt sich also sagen: Ohne das Schichtenmodell und das Copy-On-Write-Verfahren würde es Container in der heutigen Form gar nicht geben!

Container sind nichts »Wertvolles«

Vielleicht können Sie nun eher einen typischen Gedanken aus dem Container-Universum akzeptieren: Container sind nichts Wertvolles. Sie werden gestartet, gestoppt und ziemlich oft auch wieder weggeworfen. Das eigentlich Wertvolle sind vielmehr die Images (wobei wir später sehen werden: richtig wertvoll sind eigentlich auch nur die »Baupläne« für die Images) und natürlich gegebenenfalls die Daten. Speichern Sie also nie wichtige Daten in der Read-Write-Schicht eines Containers! Wenn Ihr Container sich während seiner Laufzeit wertvolle Daten erarbeitet, so müssen diese unbedingt außerhalb persistent abgelegt werden. Und genau darauf kommen wir nun endlich zu sprechen.

21.4.2 Bind Mounts und Volumes

Wie wir nun wissen, werden Datenänderungen in einem Container in seinem obersten (Read/Write-)Layer durchgeführt. Wenn der Container gelöscht wird, gehen auch all diese Änderungen verloren. Nun gibt es aber natürlich Container, die während ihrer Lebenszeit wertvolle Daten sammeln. In diesen Fällen muss unbedingt für Persistenz gesorgt werden.

Die naheliegende Idee ist dabei, ein Verzeichnis des Hosts in den Container »hineinzumounten«. `docker run` bietet hierfür zunächst einmal die Option `-v/--volume`, die es in drei Ausprägungen gibt:

- ▶ **Bind Mount**
Aufrufsyntax: `docker run -v <HOST_ABS_PATH>:<CONTAINER_ABS_PATH> [...]`
- ▶ **Named Volume**
Aufrufsyntax: `docker run -v <SIMPLE_NAME>:<CONTAINER_ABS_PATH> [...]`
- ▶ **Anonymous Volume**
Aufrufsyntax: `docker run -v <CONTAINER_ABS_PATH> [...]`

`ABS_PATH` steht dabei in allen Fällen für einen absoluten Verzeichnispfad, und `SIMPLE_NAME` ist ein einfacher Bezeichner wie »hallo« oder »test123«.

Bind Mounts

Wenn die linke Komponente des `-v`-Parameters ein absoluter Pfad ist, so wird dieses Hostobjekt im Container an der Stelle eingebunden, die durch die rechte Komponente bezeichnet wird.

```
$ docker run -it --name a1 --rm -v /tmp/data:/data alpine
```

Hierbei ist zu beachten:

- ▶ Unter Docker werden sämtliche nichtexistenten Pfade automatisch erzeugt – *sowohl auf dem Host als auch im Container!*
- ▶ Unter Podman gilt das auf der Host-Seite *nicht* – das Verzeichnis auf dem Host müsste also bereits existieren.
- ▶ Security-Module wie SELinux können verhindern, dass der Container tatsächlich Daten im gemounteten Bereich schreiben kann. Hier hilft nur, das Security-System geeignet zu konfigurieren oder abzuschalten.

Achtung beim Bind Mount normaler Dateien!

Es müssen nicht unbedingt Verzeichnisse gemountet werden. Wenn der Host-Pfad eine normale existente Datei darstellt, so wird auch nur diese Datei im Container eingebunden. Beim Bind Mount einzelner Dateien kann jedoch ein sehr subtiles Problem auftreten. Mitunter haben Sie ja die Absicht, eine solche Datei während der Laufzeit des Containers zu ändern, und zwar auf dem Hostsystem – weil es ja dort viel bequemer ist. Und der Container sollte dann sofort die neuen Inhalte sehen, weil es ja dieselbe Datei ist.

Je nach eingesetztem Editor (und dazu gehört auch `vim!`) werden Sie aber feststellen müssen, dass im Container nach einer solchen Änderung immer doch die alten Inhalte sichtbar sind! Das liegt daran, dass ein solcher Editor nicht wirklich die vorhandene Datei ändert, sondern eine neue Datei erstellt und diese an die Stelle der alten setzt. Somit entsteht im Hostsystem ein neuer Inode, aber durch das Bind Mount hält der Container nach wie vor am alten Inode fest.

Sie sollten in einem solchen Fall dann besser auf Verzeichnis-Bind-Mounts setzen, wo so etwas nicht passieren kann. Der Ratschlag, einfach einen anderen Editor (z. B. `nano`) zu benutzen, kann aus nachvollziehbaren Gründen nicht gegeben werden.

Named Volumes

Wenn die linke Komponente des `-v`-Parameters ein »simpler« Name ist, so erzeugt Docker implizit ein sogenanntes *Volume*. Dies ist ein Verzeichnis an einer bestimmten Stelle im Filesystem des Hosts. Auch hier zunächst ein Beispiel:

```
$ docker run -it --name a1 --rm -v data:/data alpine
```

Für den Container fühlt sich das genauso an wie ein Bind Mount, aber auf dem Host passiert etwas anderes:

```
$ docker volume ls
DRIVER          VOLUME NAME
local          data

$ docker volume inspect data
[
  {
    "CreatedAt": "2023-03-30T20:59:58+02:00",
    "Driver": "local",
    "Labels": null,
    "Mountpoint": "/var/lib/docker/volumes/data/_data",
    "Name": "data",
    "Options": {},
    "Scope": "local"
  }
]
```

Das `docker volume`-Kommando hat auch ansonsten ziemlich logische Subkommandos (unter Podman sind es noch einige mehr):

Usage: `docker volume COMMAND`

Commands:

```
create      Create a volume
inspect     Display detailed information on one or more volumes
ls          List volumes
prune       Remove all unused local volumes
rm          Remove one or more volumes
```

Anonymous Volumes

Anonymous Volumes entsprechen technisch den Named Volumes, nur dass Docker in dem Fall selbst einen Namen in Form einer langen, eindeutigen ID vergibt. Da diese Art von Volume schwer zuzuordnen ist, sollte sie nach Möglichkeit vermieden werden. Auch hierzu ein Beispiel:

```
$ docker run -it --name a1 --rm -v /data alpine
```

Anmerkung

Wenn ein Container mit einem Anonymous Volume *und* der Option `--rm` gestartet wird, so wird das Volume bei Beendigung des Containers automatisch wieder entfernt.



Welche Variante wählen?

Alle Varianten haben natürlich ihre Berechtigung. Bind Mounts sind für Ad-hoc- oder Entwicklungszwecke oft optimal. Aber sobald alles »etwas größer« wird, können die Named Volumes meist ihre Vorteile ausspielen:

- ▶ Sie sind einfacher in die Datensicherung einzubinden.
- ▶ Sie sind einfacher zu migrieren.
- ▶ Sie haben eine eigene Verwaltungsschnittstelle (`docker volume`).
- ▶ Es gibt verschiedene Volume-Driver zur Auswahl, um z. B. auf Remote-Hosts oder -Storage zuzugreifen. Sogar unser verwendeter `local`-Driver hat bei Bedarf schon die Möglichkeit, NFS-Freigaben einzubinden (siehe https://docs.docker.com/engine/reference/commandline/volume_create.)

21.4.3 Weitere Möglichkeiten

Volumes From

Mit dem Schalter `--volumes-from` können Sie bei der Container-Erzeugung einfach alle Volumes einbinden, die ein anderer (schon existenter) Container kennt:

```
$ docker run -it --name a2 --rm --volumes-from a1 alpine
```

Read-only-Volumes

Sie können Volumes mit dem Optionszusatz `:ro` auch Read-only einbinden:

```
$ docker run -it --name a3 --rm -v data:/data:ro alpine
```

»Neue« Aufrufsyntax

Die Option `-v` ist eigentlich schon obsolet, wird aber sicher noch ganz lange Zeit unterstützt werden. Das modernere Äquivalent ist `--mount`, und das vorige Beispiel sähe damit so aus:

```
$ docker run -it --name a3 --rm --mount type=volume,src=data,dst=/data,ro alpine
```

Siehe dazu auch <https://docs.docker.com/storage/volumes/>.

21.4.4 Informationsbeschaffung

Volumes: Was gehört wem, was kann weg?

Früher oder später haben Sie wahrscheinlich eine ganze Menge Volumes »herumliegen«, die womöglich niemand mehr braucht. Mit dem `volume prune`-Subkommando können Sie all diese Volumes löschen. Vielleicht sollten Sie aber wenigstens mal kurz schauen, was das alles für Volumes sind. Dies gelingt wie üblich mit einem Filter:

```
$ docker volume ls -f dangling=true
```

Bei den Volumes, die *nicht* »dangling« sind – die also noch mit irgendeinem Container verbunden sind –, ist manchmal nicht sofort klar, mit welchem. Hier hilft folgende Filtermöglichkeit:

```
$ docker container ls -af volume=<VOLUME_NAME>
```

Achtung

Mitunter liegt die Dangling-Suche daneben: Es werden Dangling Volumes angezeigt, die gar keine sind. Die Ursache ist unklar: Docker Compose + Docker-Live-Restore-Feature? Jedenfalls spinnt der Referenzzähler manchmal.

In jedem Falle sind Sie bitte extrem zurückhaltend, was das Löschen von Volumes angeht!

Generelle Übersicht über alle Mounts und Volumes

Mit `docker inspect` können Sie noch viel genauer in einen Container »hineinschauen«; unter `Mounts` oder `HostConfig.Binds` ist die aktuelle Mount-Situation auslesbar. Mit ein wenig `xargs`- und `Go-Template`-Magie können Sie das mehr oder weniger nett aufbereiten (danke an die `Stack-Overflow-Gemeinde!`), und am besten machen wir gleich ein Shell-Skript daraus:

```
#!/bin/bash

if docker --version 2>/dev/null 1>&2; then
    CMD=docker
else
    CMD=podman
fi

$CMD ps -aq | xargs $CMD container inspect --format \
'{{ .Name }}{{ range .Mounts }}{{ printf "\n\t%7s: " .Type -}}
{{ if eq .Type "bind" }}{{ .Source }}{{ end -}}
{{ .Name }} => {{ .Destination }}{{ end }}{{ printf "\n" }}'
```

Listing 21.5 »show_volumes.sh«: Anzeige aller Bind Mounts und Volumes

Das gibt Ihnen beim Aufruf eine Übersicht der folgenden Art:

```
/nextcloud_app
    volume: nextcloud_files => /var/www/html

/nextcloud_db
    volume: nextcloud_data => /var/lib/postgresql/data
```

```
/openldap
  bind: /root/adm/docker/openldap/certs => /certs
  bind: /root/adm/docker/openldap/ldif => /incoming/ldif
  bind: /root/adm/docker/openldap/custom => /incoming/custom
  volume: openldap_data => /bitnami/openldap
[...]
```

Was verbleibt in der Read-Write-Schicht?

Nach Möglichkeit sollten möglichst wenig Schreibzugriffe in die Read-Write-Schicht eines Containers gehen. Alle Änderungen, die ein Container während seiner Lebenszeit dort durchgeführt hat, können Sie mit `docker diff` sichtbar machen:

```
docker diff <CONTAINER> | sort
```

Dabei bedeuten:

- ▶ A
 Added
- ▶ C
 Changed
- ▶ D
 Deleted

21.5 Erstellen eigener Images mit Dockerfiles

In diesem Abschnitt möchten wir Ihnen zeigen, wie Sie eigene, an Ihre Bedürfnisse angepasste Images erzeugen können.

21.5.1 Einfaches Committen von Anpassungen

Nehmen wir einmal an, wir wollen ein angepasstes Alpine-Image erzeugen (z. B. weil jedes Mal Ihre Lieblingssoftware fehlt, wenn Sie einen Standard-Alpine-Container starten). Starten Sie dazu einen frischen Container, und nehmen Sie die gewünschten Veränderungen vor:

```
$ docker run -it --name a1 alpine
```

Paketliste aktualisieren und einige Pakete nachinstallieren:

```
/ # apk update; apk add mc tmux curl
```

```
/ # exit
```

Der Container ist damit erst einmal gestoppt. Vielleicht möchten Sie auch noch Dateien von außen in den Container bringen? Dazu steht Ihnen das Kommando `docker cp` zur Verfügung.

Rufen Sie es mit `--help` auf, die Aufrufsemantik ist völlig plausibel und einfach nachvollziehbar. Jedenfalls können Sie nun, wenn so weit alles passt, die Änderungen *committen*, was einfach bedeutet: aus dem aktuellen Stand des Containers ein Image erzeugen.

Das Stoppen wäre dazu gar nicht unbedingt nötig gewesen; beim Commit wird der Container automatisch pausiert. Eine sichere Datenkonsistenz erreichen Sie aber nur mit dem »Offline-Commit«.

```
$ docker commit \
  -m "Added some software" \
  -a user1@example.org \
  a1 localhost/myalpine:0.1
```

Unter Podman lassen Sie die Option `-m <COMMENT>` bitte weg, denn sie wird dort nicht unterstützt. Lassen Sie uns unser neues Image zusätzlich noch mit einem `latest`-Tag versehen:

```
$ docker image tag localhost/myalpine:0.1 localhost/myalpine:latest
```

Dann müsste so weit alles gut aussehen:

```
$ docker image ls
[...]
```

IMAGE	REPOSITORY	TAG	SIZE	CREATED
localhost/myalpine	0.1	5cc8270ac569	16.6MB	8 seconds ago
localhost/myalpine	latest	5cc8270ac569	16.6MB	8 seconds ago

```
[...]
```

```
$ docker history localhost/myalpine:0.1
```

IMAGE	CREATED	CREATED BY	SIZE	COMMENT
5cc8270ac569	About a minute ago	/bin/sh	12.5MB	Added some software

```
[...]
```

Und somit können wir unser neues Image verwenden:

```
$ docker run -it --rm localhost/myalpine
/ #
```

Vor- und Nachteile dieser Methode

Der Vorteil von Commits liegt auf der Hand: Ein Commit ist einfach und »ad hoc«. Im Vergleich zu den Möglichkeiten, die wir noch betrachten werden, überwiegen aber meist die Nachteile:

- ▶ (oft) keine oder wenig aussagekräftige History
- ▶ keine (automatisierte) Reproduzierbarkeit

Besonders der zweite Punkt ist in Umgebungen, in denen ernsthaft gearbeitet wird, nicht hinnehmbar. Lassen Sie uns also anschauen, wie man *reproduzierbar* Images erstellt!

21.5.2 Dockerfiles und »docker build«: Basics

Wir werden nun den Prozess des Image-Erstellens automatisieren. Der Schlüssel dazu ist ein sogenanntes *Dockerfile* – eine Steuerdatei, vergleichbar mit Makefiles oder RPM-Specfiles.

Wir wollen zunächst einmal exemplarisch dasselbe erreichen wie vorhin: ein angepasstes Alpine-Image. Erstellen wir uns eine Build-Umgebung, etwa so:

```
$ mkdir -p ~/docker/build/myalpine && cd $_
```

Und dort platzieren wir ein erstes *Dockerfile* (das exakt so heißen muss, Groß-/Kleinschreibung beachten!):

```
FROM alpine:3.7

LABEL maintainer="First User <user1@example.org>"

RUN apk update
RUN apk add mc tmux curl
```

Listing 21.6 »Dockerfile«: Ein erstes Dockerfile

Damit kann der Bau beginnen. Achten Sie auf den Punkt am Ende; `docker build` benötigt als Argument einen Pfad, den sogenannten *Build Context*. Per Default erwartet Docker dort das *Dockerfile*. Mit dem Schalter `-f` könnte man allerdings auch ein anderes Dockerfile angeben.

```
$ docker build -t localhost/myalpine:0.2 .
[...]
```

Die Angabe einer Tag-Bezeichnung mit der Option `-t` empfiehlt sich sehr, weil das Image ansonsten keinen »vernünftigen« Namen bekommt. Auch mehrere Tags mit wiederholter `-t`-Option sind möglich. Nach einem erfolgreichen Build steht das Image nun zur Verfügung. Die History unseres Images ist jetzt übrigens auch etwas interessanter geworden – werfen Sie doch noch einen Blick auf:

```
$ docker image history localhost/myalpine:0.2
```

21.5.3 Der Build-Cache und »docker build --pull«

Wenn Sie obigen Build-Prozess (`docker build [...]`) exakt noch einmal wiederholen, stellen Sie fest, dass es nun *sehr viel* schneller geht. Der Grund liegt im sogenannten *Build-Cache*.

Der Build-Prozess stellt in jedem Schritt fest, dass es nichts zu tun gibt, da im Dockerfile nichts geändert wurde. Oder andersherum gesagt: Erst *wenn* Sie einen Schritt ändern, wird *ab diesem Schritt* alles neu gebaut.

Den Build-Cache umgehen

Wie jeder Cache ist auch der Build-Cache *meistens* eine nützliche Sache – außer in den wenigen Situationen, in denen er kontraproduktiv ist. Das ist z. B. dann der Fall, wenn Sie einen Schritt wie `apk update` auf jeden Fall neu durchführen möchten, weil sich die Paketlisten geändert haben. In solchen Fällen können Sie den Build-Cache mit der Option `--no-cache` aber umgehen:

```
$ docker build --no-cache -t localhost/myalpine:0.2 .
```

Falls Sie den Build-Cache erst *ab einem gewissen Schritt* umgehen möchten, gibt es einen Trick: Sie fügen im Dockerfile *vor* besagtem Schritt eine ARG-Zeile ein:

```
ARG CACHEBUST=1
```

Listing 21.7 Ausschnitt aus einem Dockerfile

Der Name der Variablen ist dabei egal. Immer wenn Sie beim Bauen nun einen neuen, noch nie dagewesenen Wert übergeben, wird ab diesem Schritt der Cache umgangen. Garantiert neu wäre z. B.:

```
$ docker build --build-arg CACHEBUST=$(date +%s) [...]
```

»docker build --pull«

Was passiert eigentlich bei einem Build, wenn sich eines Tages das Basis-Image ändert (in unserem Fall `also alpine:3.7`)? Bei einem »normalen« Build geschieht leider gar nichts, da das Basis-Image lokal schon vorliegt und Docker dort keine Änderung feststellen kann. Falls gewünscht, können Sie hier aber die Option `--pull` anwenden, womit Docker als Erstes versuchen würde, eine gegebenenfalls aktualisierte Version des Basis-Images zu beziehen.

Falls es eine solche gibt, wird damit natürlich der komplette Build-Cache als veraltet markiert und alle Build-Schritte werden neu durchgeführt.

21.5.4 Dangling Images

Angenommen, wir möchten in unserem Image nun zusätzlich noch `vim` installieren:

```
FROM alpine:3.7
```

```
LABEL maintainer="First User <user1@example.org>"
```

```
RUN apk update
```

```
# Jetzt noch vim dazu:
```

```
RUN apk add mc tmux curl vim
```

Listing 21.8 »Dockerfile«: Leicht modifizierte zweite Version

Wir wiederholen den Build:

```
$ docker build -t localhost/myalpine:0.2 .  
[...]
```

Dank des Build-Caches wird beim Build-Prozess nur der letzte Schritt neu ausgeführt. Beachtenswert ist aber auch, dass die letzte Schicht des vorigen Images damit nun nicht mehr gebraucht (referenziert) wird. Wenn Sie also Images bauen, werden sich zwangsläufig nach und nach »verwaiste« Schichten (sogenannte *Dangling Images*) ansammeln. Sie kosten nur unnötig Plattenplatz und können wie folgt identifiziert und entfernt werden:

```
$ docker image ls -f dangling=true  
REPOSITORY TAG IMAGE ID CREATED SIZE  
<none> <none> 97f97b8d7d06 12 minutes ago 16.6MB  
$ docker image prune  
Are you sure you want to continue? [y/N] $ y  
Deleted Images:  
[...]
```

Total reclaimed space: [...]

21.5.5 Die Dockerfile-Direktiven: Ein Überblick

Wir geben nun eine Kurzübersicht über die wichtigsten Dockerfile-Direktiven (<https://docs.docker.com/engine/reference/builder/>):

- ▶ **FROM**
Angabe des Basis-Images
- ▶ **LABEL**
Metainformationen
- ▶ **RUN**
im Intermediate-Container auszuführende Aktion
- ▶ **COPY**
Datei ins Image kopieren
- ▶ **ADD**
Datei, Verzeichnis, Archiv oder Remote-Content (via URL) ins Image kopieren
- ▶ **ENV**
Umgebungsvariable für Folgeaktionen setzen
- ▶ **ARG**
Variable, die zur Build-Zeit mittels `docker build --build-args <NAME=WERT>` hereingebracht werden kann

- ▶ **EXPOSE**
Auf welchen Netzwerkports nimmt der Container Verbindungen an? (Diese Direktive dient lediglich zur Dokumentation, siehe z.B. `ExposedPorts` in `docker image inspect`.)
- ▶ **ENTRYPOINT**
Startinstruktion des Containers
- ▶ **CMD**
Default-Argument für **ENTRYPOINT** oder Standalone-Startinstruktion
- ▶ **WORKDIR**
Arbeitsverzeichnis im Container für Folgeaktionen setzen. Etwas Vorsicht: Es wird automatisch angelegt, wenn es noch nicht existiert.
- ▶ **USER**
Benutzer/Gruppen-ID für Folgeaktionen setzen
- ▶ **VOLUME**
Verzeichnis, dessen Inhalt direkt an einer definierten Stelle unterhalb von `/var/lib/docker/` abgelegt wird (und somit *nicht* im Read/Write-Layer des Containers). Man erhält dadurch ein Anonymous Volume (siehe Abschnitt 21.4.2).
- ▶ **HEALTHCHECK**
Service-Monitoring im Container

21.5.6 Ein komplexeres Beispiel mit ENV, COPY und CMD

Lassen Sie uns nun auf Basis das Standard-`httpd`-Images eine kleine dynamische Webseite mit SSI (*Server Side Includes*) realisieren. Starten wir dazu ein neues Projekt:

```
$ mkdir -p ~/docker/build/timed && cd $_
```

Zunächst das Dockerfile:

```
FROM httpd:2.4

WORKDIR /usr/local/apache2
ENV COLOR=white

RUN sed -ri \
    -e 's/^#(LoadModule .*mod_include.so)/\1/' \
    -e 's/(Options Indexes FollowSymLinks)/\1 Includes/' \
    conf/httpd.conf

RUN /bin/echo -e '\n\
AddOutputFilter INCLUDES .html\n\
SetEnv COLOR ${COLOR}\n\
' >> conf/httpd.conf
```

```
COPY index.html htdocs/
```

```
CMD ["httpd-foreground", "-e", "debug"]
```

Listing 21.9 »Dockerfile«

Und hier die (dynamische) HTML-Seite:

```
<html>
<body bgcolor='<!--#echo var="COLOR" -->'>
<h1>Date and Time: <!--#echo var="DATE_LOCAL" --></h1>
</body>
</html>
```

Listing 21.10 »index.html«

Bauen wir das Ganze, und starten wir es mit einem geeigneten Portmapping:

```
$ docker build -t localhost/timed:0.1 .
$ docker run -d --name t1 -p 80:80 --rm -e COLOR=LightGreen localhost/timed:0.1
```

Studieren Sie das *Dockerfile* bitte sehr sorgfältig – es enthält typische und wichtige Techniken!

21.5.7 CMD und/oder ENTRYPOINT

Im letzten Beispiel haben wir mit der Build-Direktive `CMD` unser Ziel erreicht, im Container automatisch einen bestimmten Prozess laufen zu lassen. Aus Sicht des Containers ist das PID Nr. 1 – also in gewisser Hinsicht auch der wichtigste Prozess.

Je nachdem, welche Dockerfile-Beispiele Sie sich ansehen, finden Sie oft stattdessen auch `ENTRYPOINT`. Manchmal sogar beides. Tragen wir ein paar Fakten zusammen, damit Sie eine Entscheidungsgrundlage bekommen, wann Sie welche Direktive einsetzen sollten:

- ▶ Sie sollten mindestens eine von beiden spezifizieren – oder vom Base-Image »erben«. Falls Sie eine geerbte Einstellung komplett loswerden möchten, so können Sie diese einfach überschreiben:

```
FROM irgendein_basis_image
```

```
CMD []
ENTRYPOINT []
```

Listing 21.11 Ausschnitt aus einem Dockerfile

- ▶ Wenn nur eine von beiden Direktiven definiert ist, so ist der *Effekt* derselbe (falls `docker run` keine zusätzlichen Parameter bekommt; siehe weiter unten).

- ▶ Für beide gibt es eine Shell- und eine Exec-Version.

- Start über `/bin/sh -c` (Shell-Form):

```
CMD      <ARG_0> <ARG_1> ... <ARG_N>
ENTRYPOINT <ARG_0> <ARG_1> ... <ARG_N>
```

- Direkter Aufruf ohne Shell (Exec-Form):

```
CMD      ["<ARG_0>", "<ARG_1>", ..., "<ARG_N>"]
ENTRYPOINT ["<ARG_0>", "<ARG_1>", ..., "<ARG_N>"]
```

Das ist nicht unwichtig, da im ersteren Fall PID 1 eben eine Shell ist. Das kann bei der Signalverarbeitung (`docker container stop` und Ähnliches) problematisch sein.

- ▶ Die Exec-Form ist in aller Regel die empfehlenswertere.
- ▶ Wenn sowohl `CMD` als auch `ENTRYPOINT` *in der Exec-Form* spezifiziert werden, dann werden die `CMD`-Argumente hinten angehängt. Dazu ein minimaler Test:

```
$ mkdir -p ~/docker/build/entrypoint-cmd && cd $_
FROM alpine
```

```
ENTRYPOINT ["/bin/echo", "Hallo"]
CMD ["unbekannter", "Besucher"]
```

Listing 21.12 »Dockerfile«: Exemplarisches Dockerfile mit `ENTRYPOINT` und `CMD`

```
$ docker build -t localhost/entrypoint-cmd-test .
$ docker run --rm localhost/entrypoint-cmd-test
Hallo unbekannter Besucher
```

- ▶ Beide können bei `docker run` überschrieben werden:

- `CMD` kann einfach durch Übergabe von Parametern überschrieben werden:

```
$ docker run --rm localhost/entrypoint-cmd-test Fritze Flink
Hallo Fritze Flink
```

- `ENTRYPOINT` kann mit der Option `--entrypoint` überschrieben werden. Allerdings kann hier nur ein einzelnes Executable angegeben werden; eine gegebenenfalls weitere Parametrisierung muss dann vollständig über `CMD`-Parameter erfolgen:

```
$ docker run --rm --entrypoint expr localhost/entrypoint-cmd-test 3 + 4
7
```

Fazit

Nehmen Sie `ENTRYPOINT` für den Teil der Kommandozeile, der sich höchstwahrscheinlich nicht ändern wird, und `CMD` für den Rest.

Wenn parametrisierte Aufrufe überhaupt kein Thema sind, dann ist es ziemlich egal, ob Sie `ENTRYPOINT` oder `CMD` verwenden, um den Hauptprozess im Container zu definieren.

21.5.8 Verwendung eigener Entrypoint-Skripte

Wenn man möchte, dass ein Container vor dem Start des eigentlichen Dienstes noch einige Dinge erledigt, kommen typischerweise *Entrypoint-Skripte* zur Anwendung.

Es gibt natürlich keinen technischen Zwang, hier unbedingt Shell-Skripte verwenden zu müssen; in der Praxis wird es aber meist gemacht. Sie benötigen daher recht tiefe Kenntnisse in der Shell-Programmierung, um solche Skripte zu analysieren, anzupassen oder gar selbst erstellen zu können. Das folgende Beispiel sollte einige Grundprinzipien verdeutlichen:

```
$ mkdir -p ~/docker/build/myalpine-entrypoint && cd $_  
FROM alpine:3.7
```

```
LABEL maintainer="First User <user1@example.org>"
```

```
RUN apk update  
RUN apk add mc tmux curl
```

```
COPY docker-entrypoint.sh /  
RUN chmod 700 /docker-entrypoint.sh
```

```
ENTRYPOINT ["/docker-entrypoint.sh"]
```

```
CMD ["/bin/sh"]
```

Listing 21.13 »Dockerfile«

```
#!/bin/sh  
  
test -n "$ANIMAL" || export ANIMAL=Elephant  
  
echo "Starting entrypoint script. Parameters: $@"  
  
IS_INITIALIZED=/CONTAINER_IS_INITIALIZED  
  
if [ ! -e $IS_INITIALIZED ]; then  
    echo "Very first startup..."  
    touch $IS_INITIALIZED  
else  
    echo "Container is already initialized..."  
fi  
  
exec "$@"
```

Listing 21.14 »docker-entrypoint.sh«

Das Bauen und Starten geschieht wie gewohnt:

```
$ docker build -t localhost/myalpine:0.4 .
$ docker run -it --name a4 --rm localhost/myalpine:0.4
Starting entrypoint script. Parameters: /bin/sh
Very first startup...
/ # $ echo $ANIMAL
Elephant
/ # $ exit
```

Mehrere Dienste in einem Container

Ein Container sollte nach Möglichkeit immer nur *für eine Sache* zuständig sein.

Sollten Sie es aber irgendwann einmal für eine gute Idee halten, dass Ihr Container mehrere Dinge »gleichzeitig« tut, dann lesen Sie bitte als Einstieg https://docs.docker.com/config/containers/multi-service_container/.

21.5.9 ».dockerignore«-Files

Bevor der Build-Prozess richtig beginnt, wird nach einer Datei namens *.dockerignore* im Wurzelordner des Build-Kontextes gesucht. Darin können Dateien oder Verzeichnisse aufgelistet werden, die auf keinen Fall per COPY oder ADD ins Image gelangen dürfen.

Im Falle von Docker könnte darüber hinaus der Docker-Dienst ja sogar auf einer anderen Maschine laufen, womit der gesamte Build-Kontext erst einmal übers Netz übertragen werden müsste. Per *.dockerignore* gelistete Dateien wären davon gleich gänzlich ausgeschlossen.

Die Syntax und Semantik von *.dockerignore*-Dateien ist an die vom Versionskontrollsystem git bekannten *.gitignore*-Dateien angelehnt:

```
# Dies ist ein Kommentar
```

```
.git/
trash.*
```

```
**/README-secret.md
```

Listing 21.15 ».dockerignore«: Eine exemplarische Ausschlussdatei

Siehe hierzu auch <https://docs.docker.com/engine/reference/builder/#dockerignore-file>.

21.5.10 Healthchecks

Mit der HEALTHCHECK-Direktive können Sie die Container-Runtime veranlassen, im Container in regelmäßigen Abständen ein gewisses Kommando zu starten und abhängig von dessen

Errorcode einen »Gesundheitsstatus« nach außen zu melden. Leider werden nur die Errorcodes 0 (»gesund«) und 1 (»ungesund«) verstanden; Abstufungen sind nicht möglich.

Auch hierzu ein Test-Projekt:

```
$ mkdir -p ~/docker/build/healthcheck && cd $_
```

Und ein kleines exemplarisches Dockerfile, in dem mithilfe von curl geprüft wird, ob der Apache noch läuft und den richtigen Content ausliefert:

```
FROM httpd:latest
```

```
RUN apt update && apt install -y curl
```

```
HEALTHCHECK CMD curl --fail -s localhost | grep -q 'It works' || exit 1
```

Listing 21.16 »Dockerfile«: Ein exemplarisches Dockerfile mit Healthcheck

Vor CMD können Sie bei Bedarf noch verschiedene Optionen angeben, wie Tabelle 21.4 zeigt.

Option	Default	Bedeutung
--interval=<DAUER>	30s	In welchen Abständen läuft der Check?
--timeout=<DAUER>	30s	Wie lange soll auf ein Check-Ergebnis gewartet werden?
--retries=<ANZAHL>	3	Nach wie vielen Fails wird unhealthy berichtet?
--start-period=<DAUER>	0s	Dem Dienst Zeit geben, um »hochzukommen«

Tabelle 21.4 Mögliche Optionen für »HEALTHCHECK«

Bauen wir das Ganze:

```
$ docker build -t localhost/myhttpd .
```



Achtung

Im Falle von Podman müssen Sie zusätzlich noch die Option `--format docker` angeben, da Podmans Standardformat *OCI* keine Healthchecks unterstützt.

Nach dem Start haben wir nun eine zusätzliche Statusanzeige:

```
$ docker run -d --name h1 --rm localhost/myhttpd
```

```
$ docker container ls -a
```

```
[...] STATUS
[...] Up 17 seconds (health: starting)
```

Und nach 30 Sekunden:

```
$ docker container ls -a
[...] Up 31 seconds (healthy)
```

Okay, machen wir es kaputt:

```
$ docker exec -it h1 sh -c "echo kaputt >htdocs/index.html"
```

Einige Zeit später sehen Sie die entsprechende Ausgabe:

```
$ docker container ls -a
[...] Up 5 minutes (unhealthy)
```

Das wird auch in den Docker-Events festgehalten:

```
$ docker events --since "1h" -f event=health_status
[...] health_status: unhealthy [...] (image=localhost/myhttpd, name=h1)
```

Sie können ganz simpel auch einfach alle ungesunden Container auflisten:

```
$ docker container ls -f health=unhealthy
```

Darüber hinaus »tun« die Healthchecks im normalen Betriebsmodus jedoch nichts. Innerhalb von Orchestrierungsumgebungen wie *Docker Swarm* oder *Kubernetes* würde der Zustand *unhealthy* – je nach Konfiguration – in der Regel jedoch einen automatischen Neustart des Containers auslösen.

21.5.11 Multistage-Builds

Seit der Docker-Version 17.05 gibt es noch ein weiteres, sehr interessantes Feature: In einem Dockerfile ist es nun möglich, mehrere FROM-Abschnitte (sogenannte *Stages*) zu spezifizieren. Und was das Ganze richtig nützlich macht: Sie können Objekte aus früheren Stages in spätere Stages hineinkopieren. Damit kann man z. B. aufwendige Build-Prozesse in frühen Stages erledigen und schließlich nur die Endprodukte in ein sehr schlankes finales Image hineingeben. Dazu betrachten wir am besten ein Beispiel:

```
$ mkdir -p ~/docker/build/hello-multistage && cd $_
```

Das exemplarische Dockerfile hat zwei Stages: In der ersten Stage wird eine Umgebung mit C-Compiler aufgesetzt, um ein kleines C-Programm zu übersetzen. Die zweite Stage erzeugt dann das eigentlich produktive Image, das neben einer Mini-Linux-Umgebung nur noch das fertige Executable enthält:

```
FROM gcc:latest AS builder
WORKDIR /root
```

```
COPY hello.c .
RUN gcc -static -o hello hello.c

FROM alpine:latest
COPY --from=builder /root/hello /usr/local/bin
CMD ["/usr/local/bin/hello"]
```

Listing 21.17 »Dockerfile«: Dockerfile mit zwei Stages

In der zweiten Stage wäre auch `FROM scratch` möglich gewesen; dann gäbe es aber natürlich kein `/usr/local/bin`, sondern nur `»/«`. In einem realen Fall käme es eben darauf an, ob man das Mini-Linux drumherum überhaupt braucht oder eben nicht. Im folgenden Listing sehen Sie nun noch den Quelltext des Programms, das aus der ersten Stage mit `COPY --from=` übernommen wird:

```
#include <stdio.h>

int main() {
    printf("Hello Docker! I like your multistage-builds...\n");
    return 0;
}
```

Listing 21.18 »hello.c«

Wir bauen und starten wie üblich mit:

```
$ docker build -t localhost/hello .
$ docker run --rm localhost/hello
```

21.5.12 Best Practices

Am Ende des Abschnitts möchten wir zusammenfassend noch ein paar Best Practices für den Image-Bau mit Dockerfiles nennen:

- ▶ Die Reihenfolge der Direktiven ist natürlich wichtig, aber insbesondere auch für das Caching. Schreiben Sie daher Dinge, die sich wahrscheinlich nicht mehr ändern, nach oben, und Dinge, die noch im Fluss sind, nach unten.
- ▶ Versuchen Sie, die Anzahl der Schichten möglichst zu minimieren. Jede Anweisung im Dockerfile erzeugt eine Schicht im finalen Image!
- ▶ Installieren Sie nichts Unnötiges.
- ▶ Ein Container sollte möglichst für eine einzige Sache zuständig sein.
- ▶ Nutzen Sie `.dockerignore`-Files, damit keine unerwünschten Dateien im Build-Prozess verarbeitet werden.
- ▶ Nutzen Sie gegebenenfalls Multistage-Builds, um schlanke, finale Images zu erzeugen.

21.6 Multi-Container-Rollout mit Docker Compose

Im letzten Abschnitt haben wir `docker build` und Dockerfiles kennengelernt, um *reproduzierbar* Images erstellen zu können – eine schöne Sache. Wenn es aber darum geht, über den Build-Prozess hinaus (kleinere) Verbünde von Containern zuzüglich Netzwerken und Volumens zu managen, brauchen wir definitiv ein weiteres Tool. An dieser Stelle hat sich *Docker Compose* etabliert, das wir uns nun anschauen wollen.

21.6.1 Installation

... unter Docker

Ursprünglich war Docker Compose eine externe Applikation, die nicht zum Kern von Docker gehörte. Das ausführbare Programm hieß `docker-compose`. Sie könnten sich auch heute noch auf <https://github.com/docker/compose/releases> ein *out of the box* lauffähiges Binary herunterladen; Sie müssen nur darauf achten, die letzte 1er-Version zu nehmen (das sollte 1.29.2 sein). Version 2 von Compose ist ein kompletter Rewrite, der nicht mehr standalone nutzbar ist, sondern nur als Plug-in für das `docker`-Kommando angeboten wird – eine lange überfällige und vernünftige Designentscheidung. Vielleicht ist Ihre Docker-Installation schon damit ausgestattet; versuchen Sie bitte einmal

```
$ docker compose version
Docker Compose version v2.17.2
```

Falls es nicht funktioniert («'compose' is not a docker command»), installieren Sie bitte folgendes Paket nach:

- ▶ **Debian, Ubuntu, CentOS, Rocky:**
`docker-compose-plugin`
- ▶ **OpenSUSE mit Zusatzrepo `Virtualization_containers`:**
`docker-compose`

Installation unter Podman

Unter Podman ist Compose leider noch nicht als Plug-in, sondern ganz *oldschool* als externe Applikation namens `podman-compose` realisiert. Im Falle einer Red-Hat-artigen Distribution empfehlen wir als bequemsten Weg die Installation aus dem bekannten EPEL-Zusatzrepo:

```
# dnf install epel-release
# dnf install podman-compose
```

Außerhalb der Red-Hat-Welt führt der allgemeine Weg über PyPI:

```
# pip3 install podman-compose
```

Wie auch immer, jetzt sollte es auch für die Arbeit mit Podman zur Verfügung stehen:

```
$ podman-compose version
[...]
using podman version: 4.4.1
podman-composer version 1.0.3
```

Und unser docker-Alias?

Der Alias `docker=podman` funktioniert in dieser Situation nicht, denn ein Alias kann ja nicht `docker compose` auf `podman-compose` umleiten. Um aber die Aufrufkompatibilität mit Docker aufrechtzuerhalten, ersetzen Sie Ihren einfachen Alias (wahrscheinlich in Ihrer `~/bashrc`) durch folgende Funktion:

```
function docker() {
    if [[ "$1" == "compose" ]]; then
        shift
        podman-compose "$@"
    else
        podman "$@";
    fi
}
```

21.6.2 Basics

Compose ist wie üblich über Subkommandos strukturiert. Durch die logische Benennung werden Sie sich hier sicher schnell heimisch fühlen. Wir geben einmal die typischsten Kommandos wieder; mit `docker compose --help` finden Sie die komplette Übersicht:

Commands:

<code>build</code>	Build or rebuild services
<code>config</code>	Validate and view the Compose file
<code>down</code>	Stop and remove containers, networks, images, ...
<code>exec</code>	Execute a command in a running container
<code>images</code>	List images
<code>kill</code>	Kill containers
<code>logs</code>	View output from containers
<code>ps</code>	List containers
<code>restart</code>	Restart services
<code>rm</code>	Remove stopped containers
<code>start</code>	Start services
<code>stop</code>	Stop services
<code>top</code>	Display the running processes
<code>up</code>	Create and start containers
<code>version</code>	Show the Docker-Compose version information

Was aber für den Moment noch wichtiger ist: Compose arbeitet mit YAML-basierten Anweisungsdateien. Sie sollten sich also zunächst mit der Sprache YAML als solcher etwas vertraut machen. (Als kleine Einführung mag z.B. Abschnitt 28.4 dienen.)

21.6.3 Ein erstes Beispiel

Per Default sucht `docker compose` im aktuellen Arbeitsverzeichnis nach einer Datei namens `docker-compose.yml`. Mit der Option `-f` kann der Pfadname einer anderen Datei angegeben werden. Beginnen wir also unser erstes Compose-Projekt:

```
$ mkdir -p ~/docker/compose/test1 && cd $_
```

Hier sehen Sie ein erstes Compose-File, das Sie unter dem Namen `docker-compose.yml` einfach dort ablegen:

```
version: "3"

services:
  alpine1:
    image: alpine:3
    stdin_open: true
    tty: true
  alpine2:
    image: alpine:3
    stdin_open: true
    tty: true
```

Listing 21.19 »docker-compose.yml«: Exemplarische erste Compose-Datei

Die Datei beginnt mit einer Versionsangabe (`version: "3"`). Diese Angabe wird benutzt, um die Kompatibilität zu den eingesetzten Compose- und Docker-Versionen sicherzustellen. Möglich sind hier die Versionen 1, 2.x und 3.x, wobei Sie Version 1 nur erreichen, indem Sie die Versionszeile *gar nicht* spezifizieren. Der Support für Version 1 wird allerdings irgendwann eingestellt, Sie sollten also möglichst darauf verzichten, jetzt noch mit Version-1-Syntax zu arbeiten. Die Unterschiede und Feinheiten der einzelnen Versionen sind ausführlich unter <https://docs.docker.com/compose/compose-file/compose-versioning/> beschrieben. Unter dem Key `services` sind nun die einzelnen Container des Szenarios beschrieben. Die dort verwendeten Direktiven sollten weitestgehend selbsterklärend sein. (Die Direktiven `stdin_open` und `tty` werden Sie später eher selten sehen; sie entsprechen `docker run -it`, was Sie ja nur für interaktive Container brauchen.)

Der Name des Verzeichnisses, in dem sich die `docker-compose.yml` befindet (in unserem Beispiel `test1`), wird bei der Benennung der Container und des neuen Bridge-Netzwerks herangezogen, das Compose standardmäßig anlegt. Wenn Sie die automatisch generierten Containernamen nicht mögen, können Sie alternativ mit der Direktive `container_name` arbeiten.

Genug der Vorrede – bringen wir das Ganze online:

```
$ docker compose up -d
[...]
Network test1_default      Created
Container test1-alpine2-1  Started
Container test1-alpine1-1  Started
```

Dabei entspricht `-d` natürlich dem *Detached Mode* von `docker run -d`. Ob alles läuft, überprüfen Sie später mit dem `ps`-Subkommando:

```
$ docker compose ps
NAME                IMAGE           COMMAND          SERVICE  [...]
test1-alpine1-1    alpine:stable  "/bin/sh"       alpine1  [...]
test1-alpine2-1    alpine:stable  "/bin/sh"       alpine2  [...]
```

Falls Sie eine Shell in einem der Container benötigen, bietet sich das Subkommando `exec` an:

```
$ docker compose exec alpine1 sh
```

Der Vorteil gegenüber `docker exec` ist hier, dass Sie mit den Compose-Containernamen arbeiten können und sich außerdem noch die Option `-it` sparen. Die gäbe es hier sowieso nicht; die TTY-Zuweisung ist in diesem Fall der Default. Wo wir gerade bei Namen sind: Die Compose-Containernamen werden im eigens angelegten Bridge-Netzwerk natürlich aufgelöst – alle beteiligten Container können sich also über ihre Namen erreichen. Mit dem `down`-Subkommando räumen Sie das ganze Szenario wieder weg:

```
$ docker compose down
```

21.6.4 Build and Run

Compose kann nicht nur Container aus fertigen Images starten, sondern bei Bedarf auch einen Build-Prozess über Dockerfiles anstoßen, wie das folgende Beispiel zeigt:

```
$ mkdir -p ~/docker/compose/test2 && cd $_
```

Sehen wir uns zunächst ein *Dockerfile* an, das nichts allzu Aufregendes zeigt:

```
FROM alpine:3.7
LABEL maintainer="First User <user1@example.org>"
RUN apk update && apk add mc tmux curl nano
```

Listing 21.20 »Dockerfile«: Ein Beispiel

Und eine passende exemplarische *docker-compose.yml*:

```
version: "3"

services:
```



```
alpine1:
  build: .
  image: local/myalpine:0.4
  stdin_open: true
  tty: true
```

Listing 21.21 »docker-compose.yml«: Ein Beispiel mit »build«-Direktiven

Die `build`-Direktive verweist dabei auf ein Verzeichnis mit einem Dockerfile darin. Mit der `image`-Direktive können Sie einen Namen für das zu erstellende Image festlegen; ansonsten würde auch hier wieder ein Name gewählt, der auf dem Projektverzeichnis basiert. Anstelle von `build <VERZEICHNIS>` wie in obigem simplem Beispiel steht Ihnen bei Bedarf auch eine ausführlichere Variante zur Verfügung:

```
build:
  context: VERZEICHNISPFAD
  dockerfile: DOCKERFILE_NAME
```

Um das ganze Setup (inklusive Build) online zu bringen, reicht auch hier ein `docker compose up -d`. In der Praxis oft nützlich sind auch folgende Aufrufe:

Nur bauen:

```
$ docker compose build
```

Dabei den Build-Cache umgehen:

```
$ docker compose build --no-cache
```

Gegebenenfalls neue Version der Basis-Images ziehen:

```
$ docker compose build --pull
```

Neu bauen und auf jeden Fall neue Container hochziehen:

```
$ docker compose up -d --build --force-recreate
```

21.6.5 Environment und Portmappings

Sehen Sie nun, wie in einer Compose-Datei Environment-Variablen und Portmappings definiert werden. Wir beziehen uns hier im `build`-Abschnitt auf das Beispiel aus Abschnitt 21.5.6 mit der »farbigen Zeitanzeige«:

```
$ mkdir -p ~/docker/compose/timed && cd $_
```

Hier sehen Sie eine exemplarische `docker-compose.yml`-Datei. Die Verwendung von `environment` und `ports` gibt sicher keine Rätsel auf:

```
version: "3"
```

```
services:
  timed:
    build: ../../build/timed
```

```

image: local/timed:0.1
ports:
  - 3700:80
environment:
  - COLOR=yellow

```

Listing 21.22 »docker-compose.yml«

**Achtung**

Dieses Beispiel ist für sich alleine nicht lauffähig!

Environment-Variablen: Ein genauerer Blick

Zur Spezifikation von Environment-Variablen, die in den Container hineingereicht werden sollen, stehen Ihnen mehrere Möglichkeiten zur Verfügung:

▶ **Variante 1:** Liste in der Compose-Datei

```

environment:
  - VAR1=wert1
  - VAR2=wert2

```

▶ **Variante 2:** Map in der Compose-Datei

```

environment:
  VAR1: wert1
  VAR2: wert2

```

▶ **Variante 3:** Verweis auf eine oder mehrere externe Dateien mit `env_file`

```

env_file: ./test1.env
### bzw.: ###
env_file:
  - ./test1.env
  - ./test2.env

```

Solche `env`-Dateien müssen dann Zeilen im `VAR=wert`-Format enthalten. Kommentarzeilen mit `#` sind in diesen Dateien ebenfalls erlaubt.

21.6.6 Volumes in Compose

Nun wollen wir noch ein simples Beispiel für die Verwendung von Volumes geben.

Sie sehen hier beim Container `alpine1`, dass ein Bind Mount einfach nur angegeben wird (und freundlicherweise hier auch ein relativer Pfad sein darf). »Echte« Docker-Volumes (siehe `alpine2`) müssen zudem unter dem Toplevel-Key `volumes:` aufgeführt werden.

```

version: "3"
services:
  alpine1:
    image: alpine
    stdin_open: true
    tty: true
    volumes:
      - ./alpine1-data:/opt/data

  alpine2:
    image: alpine
    stdin_open: true
    tty: true
    volumes:
      - data:/opt/data

volumes:
  data:

```

Listing 21.23 »docker-compose.yml«

Anmerkung

Die etwas seltsam anmutende letzte Zeile ist tatsächlich richtig. Es geht hier nur darum, Docker Compose mitzuteilen, dass `data` ein »richtiges« Named Volume ist (und kein Verzeichnisname für ein Bind Mount).

**21.6.7 Flexible Compose-Konfigurationen durch Umgebungsvariablen**

Wollen Sie Werte in der `docker-compose.yml` setzen, ohne jedes Mal die Datei zu verändern, können Sie auch hier mit Umgebungsvariablen arbeiten, z. B.:

```

[...]
  ports:
    - ${HOST_PORT}:80
[...]

```

Listing 21.24 »docker-compose.yml«: Ausschnitt – Einsatz von Umgebungsvariablen

Wenn die Variable beim Aufruf nicht gesetzt ist, würde Compose hier einen Leerstring annehmen, was aber immer von einer Warnmeldung begleitet würde:

```
$ docker compose up -d
WARNING: The HOST_PORT variable is not set. Defaulting to a blank string.
In diesem speziellen Fall folgt auch ein anschließender ERROR.
```

Linux-typisch könnten Sie die Variable beim Start auf der Kommandozeile mitgeben:

```
$ HOST_PORT=8080 docker compose up -d
```

Alternativ könnten Sie Umgebungsvariablen für Compose auch in eine `.env`-Datei im aktuellen Arbeitsverzeichnis eintragen. Vorrang haben aber (natürlich) die Shell-Umgebungsvariablen!

```
HOST_PORT=8080
```

Listing 21.25 ».env«: Eine Umgebungsvariable für docker compose

Default-Werte

Um dem lästigen Problem zu begegnen, dass der Anwender möglicherweise vergisst, benötigte Umgebungsvariablen zu setzen, können Sie mit einer speziellen Syntax auch Defaultwerte bereitstellen:

- ▶ `${VARIABLE:-DefaultWert}`
Dieser Ausdruck liefert DefaultWert, wenn VARIABLE nicht gesetzt oder leer ist.
- ▶ `${VARIABLE-DefaultWert}`
Dieser Ausdruck liefert nur dann DefaultWert, wenn VARIABLE nicht gesetzt ist.

Diese Syntax kannten Sie möglicherweise schon aus der bash-Shell. Docker Compose hat sie dort geklaut bzw. entliehen.

21.6.8 Noch mal Restart-Policys

In der Regel ist es wünschenswert, dass Compose-gesteuerte Applikationen nach einem Docker- oder Systemneustart »von alleine« wieder hochfahren. Üblicherweise kommen dazu die bereits in Abschnitt 21.2.12 besprochenen Restart-Policys zur Anwendung. Im Compose-Kontext ist die Syntax auf Ebene der einzelnen Services völlig plausibel:

```
services:
```

```
  tollerservice:  
    [...]   
    restart: unless-stopped  
    [...]
```

21.7 Betrieb und Verwendung einer eigenen Registry

Wie wir bereits wissen, nutzen unsere Container-Runtimes per Voreinstellung den Docker Hub (`docker.io`):

```
$ docker system info | grep -i registry
Registry: https://index.docker.io/v1/
```

bzw.:

```
$ podman system info | grep -A4 registries
registries:
  search:
  - registry.access.redhat.com
  - registry.redhat.io
  - docker.io
```

Als registrierter User könnten Sie auch eigene Images dorthin hochladen (»pushen«), aber es liegt auf der Hand, dass im Unternehmenseinsatz die Nutzung einer öffentlichen Registry oft keine gewollte Lösung ist. Wir wollen also im Folgenden Möglichkeiten aufzeigen, wie Sie eine eigene private »trusted« Registry aufsetzen und betreiben können, wobei unser Augenmerk hier nur auf Open-Source-Lösungen liegt.

Wichtig

Die in diesem Abschnitt beschriebenen Vorgehensweisen dienen primär dazu, die infrage kommenden Anwendungen möglichst einfach in einer Laborumgebung zur Evaluation bereitzustellen.

Detaillierte Setups für den Produktivbetrieb (gegebenenfalls noch mit Hochverfügbarkeit und Sicherheitsstrategien) zu beschreiben, ist hier ausdrücklich nicht die Intention! Sie müssen individuell prüfen, welche Lösung sinnvoll ist.



21.7.1 Vorbereitungen in einer (virtuellen) Test-/Schulungsumgebung

Damit unser Szenario etwas realistischer (und letztlich auch übersichtlicher) wird, empfehlen wir, die Registry auf einem neuen, zweiten Host zu betreiben.

Achtung

Falls Sie bislang mit Podman gearbeitet haben: Dieser neue, zweite Host soll mit der »originalen« Docker-Software ausgestattet sein, da zumindest unsere favorisierte Registry-Lösung *Harbor* dies voraussetzt.



Aufsetzen einer zweiten virtuellen Maschine

1. Wenn Sie Ihr aktuelles System klonen möchten, räumen Sie vielleicht als Erstes ein wenig auf; z. B. mit:

```
docker ps -aq | xargs -r docker stop
docker container prune -f
docker volume prune -f
docker network prune -f
docker image prune -af
```

- Überlegen Sie sich schon einmal eine IP-Adresse und einen offiziellen Namen, den Ihr geklontes System gleich bekommen soll. Machen Sie dafür einen `/etc/hosts`-Eintrag, etwa so:

```
192.168.150.100 hub.example.org
```

Listing 21.26 »/etc/hosts«: Exemplarischer Ausschnitt



Anmerkung

Wir werden den Namen `hub.example.org` ab jetzt durchgängig verwenden. Wenn Sie also davon abweichen, seien Sie bitte darauf gefasst, sehr viele Aufrufbeispiele anpassen zu müssen!

- Fahren Sie das System herunter und klonen Sie es.
- Statten Sie den Klon mit ausreichend Ressourcen aus. Diese richten sich nach der Registry-Software, die Sie einsetzen möchten.
Auf jeden Fall sollten 2 CPU-Kerne und 4 GB RAM zum Testen ausreichen. Plattenplatz wird produktiv natürlich auch ein entscheidender Faktor sein, kann in einer Testumgebung aber erst einmal vernachlässigt werden.
- Starten Sie den Klon. Setzen Sie im laufenden System einen passenden lokalen Hostnamen (z. B. »hub«) und die vereinbarte neue IP-Adresse.
- Falls Sie bislang mit Podman gearbeitet haben: Deinstallieren Sie Podman auf dem neuen Registry Host, und installieren Sie stattdessen Docker (siehe dazu die Abschnitte 21.1.6 und 21.1.8).

21.7.2 Heute mal kein TLS/HTTPS

Selbstverständlich muss der Zugriff auf eine Registry mit TLS/HTTPS abgesichert werden. Wir haben uns aber entschieden, dieses Thema an dieser Stelle weitestgehend auszuklammern und nur Verweise auf die jeweilige Online-Dokumentation anzugeben. Das hat die folgenden Gründe:

- In einer Test-/Schulungsumgebung ist die Arbeit mit Zertifikaten meist unverhältnismäßig schwierig. Man bastelt sich typischerweise z. B. mit `openssl` selbst signierte Zertifikate, die am Ende dann doch von keinem Browser und keinem Client akzeptiert werden.

Dazu müsste man mit weiteren kniffligen Vorgehensweisen, die (natürlich) für jeden Client und für jedes Betriebssystem völlig unterschiedlich sind, noch das entsprechende Stammzertifikat importieren.

In der Realität hingegen hat man typischerweise eine Zertifizierungsstelle, die »vernünftige« Zertifikate ausstellt, die *out of the box* zumindest überall in der lokalen Umgebung akzeptiert werden.

- ▶ Auch die in Betracht kommende Serversoftware müsste natürlich noch mit diesen selbst gebastelten Zertifikaten ausgestattet werden. Dabei gibt es ebenfalls oft subtile Probleme (falsches Format der Zertifikate, falscher Ablageort, falsche Zugriffsrechte, ...).

In der Realität hingegen ist das oft überhaupt nicht nötig, da z. B. ein vorgeschalteter Reverse Proxy die ganzen Probleme der Zertifikatsverteilung transparent abhandelt.

Vorweg gäbe es dann nur noch eine Kleinigkeit zu erledigen: Die Container-Runtimes möchten nicht mit unverschlüsselten (»insecure«) Registries kommunizieren – außer wir überzeugen sie davon, dass das ausnahmsweise in Ordnung ist.

Auf einem Docker-Host, der eine »insecure« Registry nutzen möchte

Für den Docker-Daemon ist eine »insecure« Registry eine HTTP-only-Registry oder eine HTTPS-Registry mit nicht validem Zertifikat. Mit folgender Konfigurationseinstellung können Sie den Daemon zur Zusammenarbeit bewegen:

```
{
  [...],
  "insecure-registries": [ "hub.example.org:5000" ]
}
```

Listing 21.27 »/etc/docker/daemon.json«: Eintrag für eine »insecure« Registry

Vergessen Sie nach dieser Konfigurationsänderung nicht den Restart des Docker-Daemons!

Es gibt übrigens keinen zwingenden Grund, Port 5000 zu verwenden. Diese Portnummer wird aber oft gewählt, da die ursprüngliche Referenzimplementierung »Docker Registry« diesen Port standardmäßig verwendet. Unsere noch folgenden Beispiele werden sich ebenfalls daran halten.

Auf einem Podman-Host, der eine »insecure« Registry nutzen möchte

Podman können Sie beispielsweise mit dieser modularen Konfigurationsdatei beruhigen:

```
[[registry]]
location = "hub.example.org:5000"
insecure = true
```

Listing 21.28 »/etc/containers/registries.conf.d/hub.example.org.conf«: Eintrag für eine »insecure« Registry

21.7.3 Harbor

Die erste Registry-Software, die wir Ihnen zeigen und auch sehr empfehlen möchten, ist Harbor (<https://goharbor.io/>). Dieses Open-Source-Projekt ist ursprünglich im Hause VMware entstanden, wurde aber mittlerweile von der *Cloud-Native Computing Foundation* (CNCF) unter die Fittiche genommen.

Test-Setup

So kommen Sie auf schnellstem Wege zu einem Test-Setup:

1. Laden Sie den vom Projekt bereitgestellten Online-Installer herunter:

```
$ wget https://github.com/goharbor/harbor/releases/download/v2.7.1/harbor-online-installer-v2.7.1.tgz
```
2. Packen Sie das Archiv aus und wechseln Sie in das entstandene Verzeichnis:

```
$ tar -xzf harbor-online-installer-v2.7.1.tgz  
$ cd harbor
```

3. Kopieren Sie die Vorlage-Konfiguration:

```
$ cp harbor.yml.tpl harbor.yml
```

4. Ändern Sie nun einige Parameter in der *harbor.yml*:

- hostname: hub.example.org
- http: / port: 5000
- Kommentieren Sie die komplette »https related config«-Sektion aus.

Es gibt noch viele weiterer interessanter Parameter, wie z.B. das Default-Admin-Passwort (*harbor_admin_password*) oder das Wurzelverzeichnis, unter dem Harbor auf dem Host seine Daten ablegt (*data_volume*). Für ein Test-Setup reicht das aber erst einmal.

5. Starten Sie nun den Installer:

```
$ ./install.sh
```

6. Wenn alle zugehörigen Container laufen, können Sie die Anwendung im Browser unter <http://hub.example.org:5000> öffnen. (Vorausgesetzt, das System, auf dem Ihr Browser läuft, kann diesen Namen auflösen – z.B. über einen */etc/hosts*-Eintrag. Ansonsten tut es auch eine IP-Adresse.)

Die Default-Zugangsdaten sind:

User: **admin**

Pass: **Harbor12345**

7. Legen Sie im obersten Menüpunkt PROJECTS ein neues Projekt an, z.B. demo1. Markieren Sie es der Einfachheit halber als *Public*.

Begeben Sie sich nun wieder auf den Host, der die Registry nutzen möchte, und testen Sie die Interaktion mit der Registry so, wie in Abschnitt 21.7.5 gezeigt.

Weitere Möglichkeiten

- ▶ Harbor hat während des Installationsprozesses u. a. eine Datei *docker-compose.yml* generiert. Damit können Sie die Applikation nun einfach mittels `docker compose` steuern.
- ▶ Sobald Sie sich für TLS/HTTPS-Setups interessieren, lesen Sie bitte den Abschnitt »Configure HTTPS Access to Harbor« unter <https://goharbor.io/docs/2.7.0/install-config/>.
- ▶ Sie können Harbor auch mit weiteren Funktionen ausstatten, z. B. mit digitalen Signaturen oder einem Vulnerability Scanner. Sie müssen dazu im Prinzip nur den Installer mit Zusatzparametern aufrufen; siehe bei Bedarf den Abschnitt »Run the Installer Script« unter <https://goharbor.io/docs/2.7.0/install-config/>.
- ▶ Sie können andere Authentifizierungsmethoden einsetzen (z. B. LDAP/AD), Benutzer mit unterschiedlichen Rechten ausstatten usw. Informationen dazu finden Sie unter <https://goharbor.io/docs/2.7.0/administration/>.

21.7.4 Docker Registry

Die Docker Registry (<https://docs.docker.com/registry/>) ist eine sehr schlanke Lösung, die wir aber trotzdem nur in Ausnahmefällen empfehlen würden, unter anderem aus folgenden Gründen:

- ▶ Es gibt keine Weboberfläche – dazu müssten Sie eine weitere Software installieren.
- ▶ Es gibt kein eingebautes Rechtemanagement – das müssten Sie sehr mühsam mit einem Reverse-Proxy »dazubasteln«.
- ▶ Es gibt keine einfache administrative Möglichkeit, Images wieder zu löschen.

Anmerkung

Wenn Sie mit der im vorigen Abschnitt vorgestellten Lösung »Harbor« zufrieden sind, dann gibt es wohl keinen Grund, sich mit der Docker Registry zu beschäftigen.



Test-Setup

So kommen Sie auf schnellstem Wege zu einem Test-Setup:

1. Legen Sie einen Compose-Projektordner an:


```
$ mkdir -p ~/docker/compose/registry && cd $_
```
2. Legen Sie dort diese Compose-Datei ab:


```
version: "3"

services:
  registry:
    image: registry:2
```

```

container_name: registry
restart: unless-stopped
environment:
  - REGISTRY_STORAGE_DELETE_ENABLED=true
  - REGISTRY_AUTH=htpasswd
  - REGISTRY_AUTH_HTPASSWD_REALM=Docker Private Registry
  - REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd
ports:
  - 5000:5000
volumes:
  - /srv/docker/registry/data:/var/lib/registry
  - /srv/docker/registry/auth:/auth

```

Listing 21.29 »docker-compose.yml«: Compose-Datei für ein Docker-Registry-Deployment

3. Legen Sie einen Ordner zzgl. einer leeren Auth-Datei an:


```
# install -d -m 700 /srv/docker/registry/auth
# install -m 600 /dev/null /srv/docker/registry/auth/htpasswd
```
4. Erzeugen Sie nun User-/Passwort-Einträge in der *htpasswd*-Datei, z. B.:


```
# docker run --entrypoint htpasswd --rm httpd:2 -Bbn user1 geheim \
  >>/srv/docker/registry/auth/htpasswd
```
5. Starten Sie den Registry-Service:


```
$ docker compose up -d
```

Begeben Sie sich nun wieder auf den Host, der die Registry nutzen möchte, und testen Sie die Interaktion mit der Registry so, wie in Abschnitt 21.7.5 gezeigt.

Weitere Möglichkeiten

- ▶ Eine TLS-Verschlüsselung erreichen Sie im Grunde recht einfach mit den zwei weiteren Umgebungsvariablen `REGISTRY_HTTP_TLS_CERTIFICATE` und `REGISTRY_HTTP_TLS_KEY` (siehe dazu <https://docs.docker.com/registry/deploying/>).
- ▶ Wie schon erwähnt, können Sie eine benutzerbezogene Rechteverwaltung nur mit einem vorgeschalteten Reverse-Proxy erlangen. Ideen dazu finden Sie unter <https://docs.docker.com/registry/recipes/>.
- ▶ Als schlankes und schlichtes Web-GUI könnte beispielsweise der *Docker Registry Browser* (<https://github.com/klausmeyer/docker-registry-browser>) von Klaus Meyer dienen. Erweitern Sie für einen schnellen Test Ihre *docker-compose.yml* um folgenden Service:

```

frontend:
  image: klausmeyer/docker-registry-browser:latest
  user: root
  restart: unless-stopped

```

```
environment:
  - DOCKER_REGISTRY_URL=http://registry:5000
  - NO_SSL_VERIFICATION=true
  - ENABLE_DELETE_IMAGES=true
  - BASIC_AUTH_USER=user1
  - BASIC_AUTH_PASSWORD=geheim
ports:
  - 8080:8080
```

Listing 21.30 Inhalt von »registry-browser-snippet.yml« für ein Docker-Registry-Deployment

Nach einem erneuten `docker compose up -d` müsste die Weboberfläche unter `http://hub.example.org:8080` erreichbar sein.

21.7.5 Arbeiten mit einer privaten Registry

Um per Kommandozeile mit einer privaten Registry zu interagieren, werden Sie sich in fast allen Fällen erst einmal authentifizieren müssen. Dazu steht das Kommando `docker login` zur Verfügung:

```
$ docker login hub.example.org:5000
Username: _____
Password: _____
[...]
Login Succeeded
```

Die Zugangsdaten werden nach erfolgreichem Login Base64-codiert in einer JSON-Datei abgelegt. Docker zeigt den Pfad zu dieser Datei von alleine an, bei Podman müssten Sie zusätzlich die `-v`-Option angeben, um diese Information zu Gesicht zu bekommen. Immerhin könnte man bei Docker stattdessen einen Credential-Helper (z. B. `pass`) verwenden. Entsprechende Rechte vorausgesetzt, könnten Sie nun beispielsweise irgendein Image in die private Registry hochladen. Dazu ist der Name des Images entscheidend, den Sie mittels `docker tag` geeignet setzen können:

Irgendein Image besorgen:

```
$ docker pull ubuntu:22.04
```

Mit einem geeigneten weiteren Namen versehen:

```
$ docker tag ubuntu:22.04 hub.example.org:5000/demo1/ubuntu:22.04
```

Hochladen ("Pushen"):

```
$ docker push hub.example.org:5000/demo1/ubuntu:22.04
```

Auch `docker pull/run` sollte natürlich mit Images aus der privaten Registry funktionieren – ebenfalls entsprechende Rechte vorausgesetzt. Wenn Sie den Registry-Zugriff nicht mehr benötigen, melden Sie sich wieder ab:

```
$ docker logout hub.example.org:5000
```


TEIL V
Kommunikation

Kapitel 22

Netzwerk

Netzwerke einzurichten und zu verwalten ist eine der wichtigsten Aufgaben eines Systemadministrators. Dabei kommt es oft zu Problemen, Fehlern und Missverständnissen. In diesem Kapitel werden wir die Untiefen der Netzwerkkonfiguration und -einrichtung unter Linux bis ins Detail beleuchten. Dabei werden wir auf die Netzwerkkonfiguration mit »ip« eingehen, das »Bonding« von Netzwerkkarten sowie das »PBR« (Policy Based Routing) und »IPv6« näher betrachten. Auch Firewalls mit »iptables« nehmen wir uns vor, bevor wir abschließend einen sehr wichtigen Dienst im Bereich Netzwerk erörtern: »DHCP«.

Nahezu alles im Linux-Bereich läuft über das Netzwerk – ob nur passiv als Client, als Server, der Dienste anbietet, oder als Teil der Netzinfrastruktur. Sogar rein lokal arbeitende Dienste laufen meist über *localhost* und werden auf dem Client auch über das Netzwerk angesprochen. Kaum eine Netzwerkfunktion kann nicht mit Linux gelöst werden. Aufgrund dieser Komplexität, der reinen Vielfalt an Möglichkeiten und der stetig steigenden Anforderungen verliert auch der ambitionierteste Systemadministrator leicht den Anschluss. In diesem Kapitel widmen wir uns den fortgeschritteneren Techniken im Netzwerkbereich.

22.1 Vorwort zu Predictable Network Interface Names

Viele Distributionen setzen mittlerweile auf *systemd*, und dabei kommt meist auch der *networkd* zum Einsatz. Diesem Umstand ist es zu verdanken, dass die sogenannten *Predictable Network Interface Names* in Linux Einzug gehalten haben. Sie sollen vor allem eine konsistente Namensgebung gewährleisten, was vorher nicht der Fall war. So konnte es vorkommen, dass beim Austausch einer defekten Netzwerkkarte alle Namen wild durcheinandergewürfelt wurden. Der *networkd* setzt zur Namensfindung folgende Methoden ein:

1. Indexnummer (Firmware/BIOS) der On-Board-Netzwerkkarten (zum Beispiel *eno1*)
2. Indexnummer (Firmware/BIOS) der Hotplug-Slot-Nummer (zum Beispiel *ens1*)
3. physischer Ort einer Netzwerkkarte (zum Beispiel *enp2s0*)
4. MAC-Adresse der Netzwerkkarte (zum Beispiel *enx78e7d1ea46da*)
5. klassische Benennung (zum Beispiel *eth0*)

Gerade der vierte Punkt führte aber zu wenig Begeisterung bei den Sysadmins. Daher wurde vereinbart, dass die Methoden zur Namensgebung wie folgt eingesetzt werden: in der Reihenfolge der Punkte 1–3, dann Punkt 5, und erst wenn alle Methoden versagt haben, greift man auf Punkt 4 zurück.



Netzwerkschnittstellen anpassen!

Aufgrund der variablen Namensgebung müssen Sie stets darauf achten, welche Netzwerkschnittstelle verwendet wird bzw. die angegebene Schnittstelle an Ihr System anpassen. Zur Vereinfachung haben wir mit den klassischen Bezeichnungen gearbeitet, also: `eth0`, `eth1`, `eth2`, `ethX`.

22.2 Netzwerkkonfiguration mit `iproute2`

Die meisten Administratoren kennen die klassischen Tools zur Netzwerkkonfiguration (`ifconfig`, `arp` und `route`) und nutzen diese regelmäßig. In vielen Fachbüchern und Artikeln werden diese Tools erwähnt und verwendet. Dabei gibt es schon seit langer Zeit ein neues Toolset, das geschaffen wurde, um die altbekannten Tools abzulösen und die neuen Möglichkeiten im Netzwerkbereich bereitzustellen: `iproute2`.

Allerdings ist `iproute2` gar nicht mal so neu. Es wurde im Zuge des Redesigns des Netzwerksystems für Kernel 2.2 entwickelt (zur Drucklegung des Buches war der Kernel 6.5 aktuell). Die Kernkomponenten von `iproute2` bilden `ip` für die IP-Konfiguration und `tc` für *traffic control*. Erst durch den Einsatz der Tools von `iproute2` können alle Netzwerkfähigkeiten von Linux voll genutzt werden.



Auch wenn Sie kein exotisches Netzwerk betreiben oder nicht unbedingt alle Funktionen ausreizen wollen, sollten Sie dennoch auf `ip` umsteigen. Insbesondere das Pendant `ifconfig` legt nämlich in einigen Situationen ein merkwürdiges Verhalten an den Tag und führt zu unerwünschten und nicht reproduzierbaren Ergebnissen.

Das Toolset `iproute2` ist schon lange Bestandteil aller Distributionen und wird in der Regel bei der Installation mit installiert. Das Paket nennt sich entweder `iproute` oder `iproute2`. In diesem Abschnitt zeigen wir Ihnen, wie Sie die altbekannten Tools mit den äquivalenten Tools von `iproute2` ablösen können.

22.2.1 Erste Schritte

Zunächst werden wir die aktuelle Konfiguration des Systems auslesen, um die Unterschiede der Tools darzustellen. Die Kommandos, die wir dafür verwenden, werden später im Detail betrachtet. Das Anzeigen der IP-Adressen erfolgt mit `ip address show`. Die Ausgabe ist dabei ganz anders als bei `ifconfig` (siehe Listing 22.1):


```

root@saturn:~# ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state \
    UP group default qlen 1000
    link/ether 08:00:27:ad:18:c3 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.163/24 brd 192.168.0.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fead:18c3/64 scope link
        valid_lft forever preferred_lft forever

```

```

root@saturn:~# ifconfig
eth0      Link encap:Ethernet  Hardware Adresse 08:00:27:ad:18:c3
          inet Adresse:192.168.0.163  Bcast:192.168.0.255  Maske:255.255.255.0
          inet6-Adresse: fe80::a00:27ff:fead:18c3/64  Gültigkeitsbereich:Verbindung
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metrik:1
          RX packets:24778 errors:0 dropped:166 overruns:0 frame:0
          TX packets:7496 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:2765211 (2.6 MiB)  TX bytes:1270890 (1.2 MiB)

lo        Link encap:Lokale Schleife
          inet Adresse:127.0.0.1  Maske:255.0.0.0
          inet6-Adresse: ::1/128  Gültigkeitsbereich:Maschine
          UP LOOPBACK RUNNING  MTU:65536  Metrik:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:0
          RX bytes:1053 (1.0 KiB)  TX bytes:1053 (1.0 KiB)

```

Listing 22.1 Übersicht über die IP-Adressen mit »ip« und »ifconfig«

Mit ip können Sie auch Routen auslesen und setzen. Daher dient es auch zur Ablösung von route. Routing-Tabellen werden mit ip route angesprochen (siehe Listing 22.2):

```

root@saturn:~# ip route show
default via 192.168.0.1 dev eth0
192.168.0.0/24 dev eth0  proto kernel scope link src 192.168.0.163

```

Listing 22.2 Übersicht über die Routing-Tabellen mit »ip«

Zum Vergleich haben wir die Ausgabe von `route` in Listing 22.3 dargestellt:

```
root@saturn:~# route -n
Kernel-IP-Routentabelle
Ziel          Router        Genmask       Flags Metric Ref    Use Iface
0.0.0.0       192.168.0.1  0.0.0.0       UG    0      0      0 eth0
192.168.0.0   0.0.0.0      255.255.255.0 U     0      0      0 eth0
```

Listing 22.3 Übersicht über die Routing-Tabellen mit »route«

Auch das Kommando `arp` hat einen Nachfolger: `ip neighbour`. Listing 22.4 zeigt die Ausgabe:

```
root@saturn:~# ip neighbour show
192.168.0.1 dev eth0 lladdr 20:c9:d0:13:0d:72 STALE
192.168.0.105 dev eth0 FAILED
192.168.0.20 dev eth0 lladdr 20:c9:d0:13:0d:72 DELAY
```

Listing 22.4 Anzeigen der »arp«-Tabelle mit »ip neighbour«

Auch hier unterscheidet sich die Ausgabe zum Vorgänger `arp` (siehe Listing 22.5):

```
root@saturn:~# arp -na
? (192.168.0.1) auf 20:c9:d0:13:0d:72 [ether] auf eth0
? (192.168.0.105) auf <unvollständig> auf eth0
? (192.168.0.20) auf 20:c9:d0:13:0d:72 [ether] auf eth0
```

Listing 22.5 Anzeigen der »arp«-Tabelle mit »arp«

Im Gegensatz zu `ifconfig` unterscheidet `ip` generell zwischen Links und Adressen. Dank dieser Neuerung kann der Zustand der Links auch separat angezeigt werden, wie in Listing 22.6 dargestellt:

```
root@saturn:~# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP\
    mode DEFAULT group default qlen 1000
    link/ether 08:00:27:17:36:61 brd ff:ff:ff:ff:ff:ff
```

Listing 22.6 Übersicht über die »link«-Einträge mit »ip«

Die Ausgaben des alten und des neuen Toolsets unterscheiden sich erheblich voneinander. Auch die Darstellungsform ist sehr gewöhnungsbedürftig und stellenweise etwas unübersichtlich. Nichtsdestotrotz lohnt sich der Umstieg, da einige Informationen von den alten Tools nämlich schlichtweg unterschlagen werden. Dies werden wir in den folgenden Abschnitten näher betrachten.

22.2.2 Die Syntax von ip

Die Syntax von `ip` ist im Gegensatz zu den altbekannten Tools deutlich komplexer, dafür ist sie aber vereinheitlicht. Das Programm `ip` erwartet als ersten Parameter immer ein Objekt, mit dem definiert wird, welcher Bereich angesprochen wird. Beispielsweise nutzen Sie `ip address` zum Konfigurieren von IP-Adressen oder `ip route`, wenn die Routing-Tabelle manipuliert werden soll.

Je nach Objekt können dann unterschiedliche Kommandos ausgeführt werden. Wird kein Kommando angegeben, so wird einfach `show` oder `list` ausgeführt. Dabei ist `ip` äußerst intelligent. Alle Kommandos lassen sich abkürzen, solange eine Anweisung eindeutig von anderen unterscheidbar ist. Daher führen folgende Kommandos alle zum gleichen Ergebnis (hier zur Ausgabe der IP-Adresskonfiguration):

- ▶ `ip address show`
- ▶ `ip add sh`
- ▶ `ip a s`
- ▶ `ip a`

22.2.3 Links ansehen und manipulieren: ip link

Informationen über Links können mit `ip link show <DEVICE>` ausgegeben werden, so wie es in Listing 22.7 dargestellt ist:

```
root@saturn:~# ip link show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP\
    mode DEFAULT group default qlen 1000
    link/ether 08:00:27:17:36:61 brd ff:ff:ff:ff:ff:ff
```

Listing 22.7 Informationen der Netzwerkkarte mit »ip« abrufen

In der ersten Zeile gibt `ip` die Nummer des Interface 2 aus, gefolgt von dem eindeutigen Namen `eth0`. Danach folgen diverse Flags, die die Fähigkeiten und den Zustand der Schnittstelle beschreiben. Die wichtigsten Flags sind:

- ▶ `UP`
Die Schnittstelle ist aktiviert.
- ▶ `LOWER_UP` und `NO-CARRIER`
Dieses Flag zeigt an, ob eine Verbindung auf Ethernet-Ebene besteht. Die Flags werden nur angezeigt, wenn das Interface auch aktiviert ist.
- ▶ `BROADCAST` und `MULTICAST`
zeigt an, ob das Interface Broad- und Multicast beherrscht.

- ▶ SLAVE
gibt an, dass die Schnittstelle zu einem Bonding (Netzwerkkartenbündelung) gehört.
- ▶ POINTOPOINT
gibt an, dass es sich um eine Punkt-zu-Punkt-Verbindung handelt. Das Netzwerk besteht nur aus zwei Knoten. Jeder Verkehr landet automatisch bei der Gegenstelle.

Nach den Flags wird die *MTU*-Größe angezeigt (`mtu <SIZE>`), die *Maximum Transmission Unit*. Sie gibt die maximale Paketgröße auf der Sicherungsschicht an. Bei Ethernet sind das 1.500 Byte. Das Loopback-Interface arbeitet hingegen mit 16.436 Byte, weil es als virtuelles Interface nicht den Beschränkungen von Ethernet unterliegt. Andere Größen für die *MTU* kommen regelmäßig bei Tunneln vor, da die Pakete dort gekapselt werden. Danach folgt die Angabe der *Queuing Discipline* (`qdisc`), gefolgt von deren Algorithmus (`pfifo_fast`) für das Handling der Pakete und der Queue-Länge (`qlen 1000`). In der zweiten Zeile stehen Informationen über den Linktyp und linkspezifische Informationen. Bei Ethernet sind das die MAC-Adresse und die Ethernet-Broadcast-Adresse.

Auch zusätzliche Paketstatistiken rund um die Verbindungen können mit `ip link` angezeigt werden. Verwenden Sie dafür die Option `-statistics` bzw. `-s`. Die Option kann mehrfach angegeben werden, um den Detailgrad zu erhöhen (siehe Listing 22.8):

```
root@saturn:~# ip -s -s link show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP\
    mode DEFAULT group default qlen 1000
    link/ether 08:00:27:ad:18:c3 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun  mcast
    92329     1131     0       28       0         167
    RX errors: length  crc      frame   fifo     missed
                  0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier  collsns
    7391      57       0       0       0         0
    TX errors: aborted fifo    window heartbeat transns
                  0       0       0       0         2
```

Listing 22.8 Paketstatistiken

Die Parameter von Links werden mit `ip link set <DEVICE>` geändert. Neben der Aktivierung und Deaktivierung von Schnittstellen können Sie auch Flags, MAC-Adressen oder sogar Devisenamen ändern. Die entsprechenden Befehle wollen wir Ihnen kurz vorstellen.

Interface deaktivieren und aktivieren

```
root@saturn:~# ip link set eth0 down
root@saturn:~# ip link set eth0 up
```

Listing 22.9 Aktivieren und Deaktivieren einer Netzwerkkarte

Neue MAC-Adresse setzen

```
root@saturn:~# ip link set eth0 address 00:26:18:81:1d:65
```

Listing 22.10 Setzen der MAC-Adresse

Interfacenamen ändern

```
root@saturn:~# ip link set eth1 name extern
root@saturn:~# ip link show extern
6: extern: <BROADCAST,NOARP> mtu 1500 qdisc noqueue state DOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
```

Listing 22.11 Ändern des Interfacenamens

Den Namen von Netzwerkkarten zu ändern kann bei Setups mit vielen Netzwerkkarten durchaus Sinn machen, beispielsweise bei Firewalls, um direkt über den Namen anzugeben, welches Interface wohin führt (intern, extern, dmz, management, heartbeat, vpn ...).

Deaktivieren Sie vor Änderungen das entsprechende Interface

Ändern Sie Namen von Interfaces nur, wenn das Interface deaktiviert ist. Ansonsten kommen die laufenden Dienste durcheinander, die bereits auf dem Interface gebunden sind! In den aktuellen Versionen wird dies aber direkt von ip unterbunden – Sie bekommen eine entsprechende Fehlermeldung, wenn Sie versuchen, Namen von aktiven Interfaces zu ändern:

```
root@saturn:~# ip link set eth1 name extern
RTNETLINK answers: Device or resource busy
```

Effektiver, und vor allem auch bootresistent, ändern Sie die Namen von Interfaces durch das Erstellen einer *udev*-Regel (siehe Abschnitt 3.3.1, »udev-Regeln«).



22.2.4 IP-Adressen ansehen und manipulieren: ip address

Die Ausgabe der aktiven IP-Adresskonfiguration erfolgt, wie bereits erörtert wurde, mittels `ip address show`. Zusätzlich zum Link, auf dem die IP-Adressen konfiguriert sind, werden auch weitere Informationen ausgegeben. Listing 22.12 zeigt eine typische Ausgabe:

```
root@saturn:~# ip address show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP \
    group default qlen 1000
    link/ether 08:00:27:17:36:61 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.163/24 brd 192.168.0.255 scope global eth0
    inet6 fe80::a00:27ff:fe8a:a89f/64 scope link
        valid_lft forever preferred_lft forever
```

Listing 22.12 Anzeige der IP-Konfiguration

An erster Stelle sehen Sie das genutzte Protokoll. In Listing 22.12 sind das `inet` und `inet6` für `ipv4` bzw. `ipv6`. Danach folgen die IP-Adresse in *CIDR-Notation* und die Broadcast-Adresse. Bei Punkt-zu-Punkt-Verbindungen wird anstelle der Broadcast-Adresse die IP-Adresse der Gegenseite angezeigt. Mit `scope` wird der Geltungsbereich angezeigt. Mögliche Werte von `scope` sind:

- ▶ `host`
Die konfigurierte IP-Adresse ist nur auf dem lokalen Host gültig und außerhalb unbekannt. Dies gilt zum Beispiel für die Adresse `127.0.0.1` auf dem Loopback-Device.
- ▶ `global`
Die Adresse hat eine globale Gültigkeit (Standardwert für normale IP-Adressen).
- ▶ `link`
Die Adresse ist nur auf dem gezeigten Link gültig. Das heißt, sie wird nur zur Kommunikation im LAN eingesetzt und wird nicht geroutet. Das ist beispielsweise für die Adressen `169.254.0.0/16` der Fall, die von *APIPA (Automatic Private IP Addressing)* genutzt werden.
- ▶ `site`
Diese Adressen sind nur in der aktuellen Site gültig und werden von Routern nicht nach außen geroutet oder von außen akzeptiert. Dieser Scope-Typ gilt nur für *IPv6* und bildet private Netze unter *IPv6*.
- ▶ `secondary`
Danach kann noch `secondary` stehen, das eine sekundäre IP-Adresse spezifiziert. Auf diese Funktion gehen wir bei der IP-Adresskonfiguration näher ein.
- ▶ `interfacelabel`
Nach `secondary` oder `scope` folgt das Interface-Label. Dieses Label ist entweder nur der Interface-Name oder der Interface-Name, gefolgt von einem Doppelpunkt und einem beliebigen Zusatznamen. Dieses Label kennen Sie vom Konfigurieren zusätzlicher IP-Adressen per `ifconfig` (zum Beispiel `eth0:1`).

Listing 22.13 zeigt, wie Sie die laufende IP-Konfiguration mit `ifconfig` ändern können:

```
root@saturn:~# ifconfig eth1:0 192.168.250.1 netmask 255.255.255.0 up
root@saturn:~# ip address show dev eth1
3: eth1: <BROADCAST,NOARP> mtu 1500 qdisc noqueue state DOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global eth1
    inet 192.168.250.1/24 brd 192.168.250.255 scope global eth1:0
    inet6 fe80::a00:27ff:fe05:6dc4/64 scope link
```

Listing 22.13 Konfiguration der Netzwerkkarte mit »`ifconfig`«

Nur mit `ifconfig` allein können Sie sich nie sicher sein, ob tatsächlich alle IP-Adressen angezeigt werden. Daher haben wir für Sie ein Beispiel für nicht angezeigte IP-Adressen erstellt.

Die zweite IP-Adresse, 192.168.201.1, ist per `ifconfig` nicht zu sehen, wie es in Listing 22.14 dargestellt ist:

```
root@saturn:~# ifconfig eth1
eth1  Link encap:Ethernet Hardware Adresse 08:00:27:05:6d:c4
      inet Adresse:192.168.200.1 Bcast:192.168.200.255 Maske:255.255.255.0
      BROADCAST NOARP MTU:1500 Metrik:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
      Kollisionen:0 Sendewarteschlangenlänge:0
      RX bytes:0 (0.0 B) TX bytes:980 (980.0 B)
root@saturn:~# ip address show eth1
3: eth1: <BROADCAST,NOARP> mtu 1500 qdisc noqueue state DOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global eth1
    inet 192.168.201.1/24 brd 192.168.201.255 scope global eth1
```

Listing 22.14 Vergleich der Anzeige von »ifconfig« und »ip«

IP-Adressen ohne Label können mit »ifconfig« nicht mehr angezeigt werden

Mit `ip` konfigurierte IP-Adressen benötigen kein Label – da Label optional sind. Das Programm `ifconfig` zeigt aber nur Interfaces mit gesetztem Label an. Das allein spricht schon für die Nutzung von `ip`.



Das Hinzufügen oder Löschen von IP-Adressen erfolgt mit `ip address add` bzw. `ip address delete`. Selbstverständlich können Sie auch diese Befehle abkürzen, zum Beispiel zum Hinzufügen mit `ip a a` und zum Löschen mit `ip a d`. Die IP-Adresse wird immer in CIDR-Notation angegeben. Lässt man das Suffix weg, so erhält man eine Hostadresse (/32-Adresse).

Leider wird die Broadcast-Adresse nicht automatisch gesetzt und muss mit angegeben werden – was durchaus korrekt ist, da Broadcast nicht zwingend für ein Netz vorausgesetzt ist. Die Angabe einer Broadcast-Adresse geschieht mit `broadcast ADDRESS`. Wahlweise können Sie statt der ausgeschriebenen Adresse auch `+` oder `-` angeben, um einfach alle Hostbits auf 1 oder 0 zu setzen, was die Angabe deutlich vereinfacht.

Zusätzlich muss mit `dev <DEVICE>` immer angegeben werden, auf welcher Schnittstelle die IP-Adresse gelöscht oder hinzugefügt werden soll. Dort verlässt sich `ip` nicht auf seine Intelligenz, sondern möchte diese Information von Ihnen als Administrator übergeben bekommen. Listing 22.15 zeigt das Hinzufügen und Löschen mittels `ip`:

```
root@saturn:~# ip address add 192.168.99.14/24 broadcast + dev eth1
root@saturn:~# ip address delete 10.10.10.1/24 dev eth1
```

Listing 22.15 Hinzufügen und Löschen einer IP-Adresse mit »ip«

Zum bequemen Löschen aller IPv4-Adressen eines Interface können Sie den Befehl `flush` nutzen. Listing 22.16 zeigt, wie Sie alle IPv4-Adressen vom Interface `eth1` entfernen können:

```
root@saturn:~# ip -4 address flush label eth1
```

Listing 22.16 Löschen aller IPv4-Adressen mit »flush«

Oft werden mehrere Adressen auf ein Interface gebunden, um zum Beispiel einem Server pro Dienst eine IP-Adresse zu spendieren. In diesem Fall gibt es einige Besonderheiten, die Sie beachten müssen.

Wenn mehrere IP-Adressen aus dem gleichen Netz (das heißt mit dem gleichen Netzwerkpräfix) auf ein Interface gebunden werden, so ist die erste IP-Adresse eine primäre und die folgenden Adressen sind die sekundären. In Listing 22.17 werden mehrere IP-Adressen aus dem Subnetz `192.168.200.0/24` dem Interface `eth1` zugeordnet, und anschließend wird die Konfiguration mittels `ip` angezeigt:

```
root@saturn:~# ip a a 192.168.200.1/24 brd + dev eth1
root@saturn:~# ip a a 192.168.200.2/24 brd + dev eth1
root@saturn:~# ip a a 192.168.200.3/24 brd + dev eth1
root@saturn:~# ip a s eth1
3: eth1: <BROADCAST,NOARP> mtu 1500 qdisc noqueue state DOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global eth1
    inet 192.168.200.2/24 brd 192.168.200.255 scope global secondary eth1
    inet 192.168.200.3/24 brd 192.168.200.255 scope global secondary eth1
```

Listing 22.17 Mehrere IP-Adressen aus einem Subnetz

In diesem Beispiel ist `192.168.200.1` die primäre IP-Adresse. Im Normalfall wird nur diese IP-Adresse für ausgehende Pakete des Netzes `192.168.200.0/24` verwendet. Selbstverständlich werden Antworten auf Anfragen, die sich an sekundäre Adressen richten, auch von diesen beantwortet (ansonsten wären inkonsistente Sessions die Folge).

Jede neue IP-Adresse aus einem neuen Netzbereich wird wieder zu einer primären Adresse. Um dies zu verdeutlichen, haben wir in Listing 22.18 zwei weitere IP-Adressen dem Interface `eth1` hinzugefügt – diesmal aus dem Subnetz `192.168.250.0/24`:

```
root@saturn:~# ip a a 192.168.250.1/24 brd + dev eth1
root@saturn:~# ip a a 192.168.250.2/24 brd + dev eth1
root@saturn:~# ip a s eth1
3: eth1: <BROADCAST,NOARP> mtu 1500 qdisc noqueue state DOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global eth1
    inet 192.168.250.1/24 brd 192.168.250.255 scope global eth1
```



```
inet 192.168.200.2/24 brd 192.168.200.255 scope global secondary eth1
inet 192.168.200.3/24 brd 192.168.200.255 scope global secondary eth1
inet 192.168.250.2/24 brd 192.168.250.255 scope global secondary eth1
```

Listing 22.18 Zusätzliche primäre IP-Adressen hinzufügen

22

Dank der Intelligenz von `ip` wurden diese Aufgaben korrekt umgesetzt. Nun kommen wir zu den angesprochenen Besonderheiten. Wird eine primäre Adresse gelöscht, so werden auch alle dazugehörigen sekundären Adressen gelöscht (siehe Listing 22.19):

```
root@saturn:~# ip a d 192.168.200.1/24 dev eth1
root@saturn:~# ip a s eth1
3: eth1: <BROADCAST,NOARP> mtu 1500 qdisc noop state DOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.250.1/24 brd 192.168.250.255 scope global eth1
    inet 192.168.250.2/24 brd 192.168.250.255 scope global secondary eth1
```

Listing 22.19 Löschen aller sekundären IP-Adressen

Achten Sie daher beim Entfernen von IP-Adressen sorgfältig darauf, nicht mehr zu entfernen, als Sie eigentlich wollen.

22.2.5 Manipulation von ARP-Einträgen: `ip neighbour`

Mit `ip neighbour show` (kurz `ip n s`) lässt sich die ARP-Tabelle so anzeigen, wie es in Listing 22.20 dargestellt ist:

```
root@saturn:~# ip n s
192.168.0.1 dev eth0 lladdr 20:c9:d0:13:0d:72 REACHABLE
192.168.0.20 dev eth0 lladdr 10:dd:b1:e4:04:f4 DELAY
192.168.0.11 dev eth0 FAILED
192.168.0.38 dev eth0 INCOMPLETE
```

Listing 22.20 Auflisten der ARP-Tabelle mit »ip«

Wie Sie in Listing 22.20 sehen, zeigt die Ausgabe, welche IP-Adresse über welches Device unter welcher MAC-Adresse erreichbar ist. Die letzte Spalte enthält den Zustand, die sogenannte NUD (*Neighbour Unreachability Detection*). Mögliche Werte sind dabei:

- ▶ REACHABLE
Der Nachbar ist erreichbar, und der Eintrag ist gültig. Die Einträge werden nach einem Timeout gelöscht.
- ▶ STALE
Der Eintrag war gültig, ist aber nicht mehr aktuell. STALE ist der Folgestatus von REACHABLE. Wenn der Eintrag nicht erneut angefragt wird, verschwindet er nach dem Ablauf des Timeouts aus der Liste.

- ▶ DELAY
Der angefragte Eintrag war zuvor in einem STALE-Zustand. Der Kernel fragt daher direkt bei dem Zielhost an, um die Adresse zu verifizieren.
- ▶ PROBE
Nach Ablauf des Delay-Timeouts kommt der Eintrag in den PROBE-Status, wenn der Kernel beim Zielsystem angefragt und noch keine Antwort erhalten hat. Daraufhin wird eine erneute Auflösung per ARP-Request angestoßen.
- ▶ FAILED
Die angefragte Adresse ist nicht im Netz ermittelbar.
- ▶ INCOMPLETE
Die Antwort auf den gesendeten ARP-Request steht noch aus.
- ▶ PERMANENT
Der Eintrag ist immer gültig.

Um Einträge zu entfernen, verwenden Sie den Befehl `ip neighbour delete`. Dieser Befehl erwartet sowohl die IP-Adresse als auch das Device als Parameter:

```
root@saturn:~# ip neighbour delete 192.168.2.1 dev eth0
```

Listing 22.21 ARP-Einträge löschen

Zum Hinzufügen und Ändern von ARP-Einträgen können Sie `ip neighbour add`, `change` und `replace` verwenden. Teilweise lässt sich auch die NUD manipulieren.

So können Sie zum Beispiel die MAC-Adresse eines Routers aus Sicherheitsgründen festlegen (siehe Listing 22.22):

```
root@saturn:~# ip neighbour change 192.168.0.1 dev eth0 lladdr \  
                20:c9:d0:13:0d:72 nud permanent  
root@saturn:~# ip neighbour show  
192.168.2.1 dev eth0 lladdr 20:c9:d0:13:0d:72 PERMANENT
```

Listing 22.22 Manipulation der »NUD«

Über den Befehl aus Listing 22.22 wird die MAC-Adresse des Systems mit der IP-Adresse 192.168.0.1 dauerhaft gesetzt. Permanente Einträge können durch das Setzen auf `reachable` oder `stale` wieder geändert werden. Anschließend unterliegen diese Einträge wieder dem normalen Timeout.



Groß- und Kleinschreibung beachten!

Beim Anzeigen der ARP-Tabelle wird die NUD häufig in Großbuchstaben ausgegeben. Beim Anpassen erwartet das Kommando die NUD aber in Kleinbuchstaben!

Auch für ARP-Einträge können Sie Statistiken mit dem Schalter `-s` abrufen. Die Ausgabe wird dann deutlich umfangreicher:

```
daniel@saturn:~$ ip -s nei
192.168.0.1 dev eth0 lladdr 20:c9:d0:13:0d:72 ref 1 used 18/18/13 probes 1 REACHABLE
192.168.0.20 dev eth0 lladdr 10:dd:b1:e4:04:f4 ref 1 used 51/0/50 probes 0 REACHABLE
192.168.0.105 dev eth0 lladdr 08:00:27:eb:fb:b3 used 145/145/122 probes 4 STALE
192.168.0.21 dev eth0 used 37/37/34 probes 6 FAILED
```

Listing 22.23 Statistik zu den Nachbarn

Wie Sie Listing 22.23 entnehmen können, sind nun auch zusätzliche Informationen enthalten, zum Beispiel werden nun Zahlwerte im Format `<VALUE>/<VALUE>/<VALUE>` zu den Einträgen ausgegeben. Dabei bedeuten die Werte zuletzt benutzte, zuletzt bestätigte und zuletzt aktualisierte ARP-Anfragen.

22.3 Routing mit ip

Mit `ip` können Sie das volle Routing-Potenzial von Linux ausnutzen, was mit dem alten Tool `route` nicht möglich war. Das Routing ist daher einer der Hauptgründe für die Nutzung von `iproute2`.

Aufgrund der gestiegenen Komplexität des ohnehin schon nicht ganz trivialen Themas wird im folgenden Abschnitt nur die »normale« Routing-Funktionalität besprochen. In Abschnitt 22.3.2, »Da geht noch mehr: ›Advanced Routing‹«, lernen Sie dann ein erweitertes Routing-Modell kennen.

22.3.1 Routing-Informationen anzeigen

Das Kommando `ip route show` (oder kurz `ip r`) gibt Ihnen die aktuelle Routing-Tabelle aus:

```
root@saturn:~# ip route show
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.163 metric 1
192.168.200.0/24 via 192.168.0.2 dev eth0
169.254.0.0/16 dev eth0 scope link metric 1000
default via 192.168.0.1 dev eth0 proto static
```

Listing 22.24 Anzeige der Routing-Tabelle

Pro Zeile wird eine Routing-Information dargestellt. Sehen wir uns die erste Zeile der Ausgabe in Listing 22.24 genauer an. Zunächst wird das Routing-Ziel angegeben, im Beispiel `192.168.0.0/24`. Danach folgt der *routing protocol identifier*. Dieser gibt an, wie die Route in die Tabelle gelangt ist – im Beispiel `proto kernel`, was angibt, dass der Kernel die Route angelegt hat. Der Kernel hat diese Route anhand der Interface-Konfiguration erstellt (`192.168.0.163/24`).

Diese Route wird auch als *directly connected* bezeichnet, da das Netzwerk direkt am Interface anliegt.

Anstelle von `proto kernel` könnte als *routing protocol identifier* auch `static` (für statisch eingetragene Routen), `redirect` (für durch *ICMP-Redirects* eingetragene Routen) oder der Protokoll- bzw. Programmname stehen, über den die Route an den Kernel gemeldet wurde. In der Datei `/etc/iproute2/rt_protos` finden Sie alle möglichen *routing protocol identifiers*. Ein *routing protocol identifier* muss nicht angegeben werden, wie Sie an den übrigen Zeilen aus Listing 22.24 sehen können.

Anschließend folgt der `scope`, der die Reichweite bzw. die Entfernung zum angegebenen Routing-Ziel angibt. In der ersten Zeile befindet sich an dieser Stelle `link`, da (wie bereits erörtert) das Netzwerk direkt am Interface anliegt. Auch hier können wieder unterschiedliche Werte angegeben sein, zum Beispiel `global` für Ziele, die über einen Router erreicht werden, oder `local` für Ziele, die lokal erreicht werden können. Auch diese Angabe ist nicht zwingend erforderlich, da sich der Wert im Normalfall aus dem Ziel ergibt.

Die nächste Spalte, `src 192.168.0.163`, gibt an, von welcher Adresse aus Pakete an das Routing-Ziel gesendet werden. Dabei ist die angegebene IP-Adresse normalerweise die primäre IP-Adresse des Systems aus dem Netzbereich.

Der letzte Wert stellt die Metrik dar (im Beispiel `metric 1`). Auch dieser Wert ist optional und gibt die Präferenz der Route an. Dies ist vor allem bei dynamischen Routings relevant, da dort anhand der Metrik errechnet wird, welche Route die günstigste ist. Das gleiche Verfahren wird im Übrigen auch angewandt, wenn zwei ansonsten gleiche Routen zu einem Routing-Ziel existieren. Bei dem Verfahren gilt: je kleiner der Wert, desto günstiger die Route.

Eine weitere Art von Routing-Einträgen sehen Sie bei der zweiten und vierten Route. Mit dem Schlagwort `via` wird festgelegt, dass das Routing-Ziel ein Router ist. Bei der zweiten Route wird definiert, dass das Netz `192.168.200.0/24` über den Router `192.168.0.2` erreichbar ist. Als Interface dafür wird `eth0` verwendet. Die vierte und letzte Route gibt das *Default-Gateway* an – anstelle des Schlagworts `default` könnte dort auch `0.0.0.0/0` stehen.

Das Hinzufügen und Entfernen von Routen kann (wie bei den bisherigen `ip`-Befehlen auch) durch Angabe von `add` und `delete` erreicht werden. Beim Hinzufügen von Routen wird mit dem Schlagwort `via` das Gateway angegeben, über das das Netz erreicht werden kann (siehe Listing 22.25). Optional können Sie dabei auch das Interface angeben.

```
root@saturn:~# ip route add 192.168.1.0/24 via 192.168.0.2
root@saturn:~# ip route add 192.168.3.0/24 dev eth1 via 192.168.250.1
root@saturn:~#
root@saturn:~# ip r s
default via 192.168.0.1 dev eth0
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.163
192.168.1.0/24 via 192.168.0.2 dev eth0
```

```
192.168.3.0/24 via 192.168.250.1 dev eth1
192.168.250.0/24 dev eth1 proto kernel scope link src 192.168.250.2
169.254.0.0/16 dev eth0 scope link metric 1000
```

Listing 22.25 Anlegen einer Route mit »ip«

Ebenso funktioniert das Löschen von Routen:

```
root@saturn:~# ip route delete 192.168.2.0/24
```

Listing 22.26 Löschen einer Route mit »ip«

Wird eine IP-Adresse entfernt, so werden die mit ihr assoziierten Routen ebenfalls gelöscht:

```
root@saturn:~# ip a d 192.168.250.2/24 dev eth1
root@saturn:~#
root@saturn:~# ip r s
default via 192.168.0.1 dev eth0
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.163
192.168.1.0/24 via 192.168.0.2 dev eth0
169.254.0.0/16 dev eth0 scope link metric 1000
```

Listing 22.27 Entfernen einer IP-Adresse und der dazugehörigen Routen

Nach Veränderungen an der Routing-Tabelle sollten Sie den Routing-Cache leeren. Die Routing-Tabelle wird nur gefragt, falls noch kein Eintrag im Cache vorhanden ist. Hierfür können Sie den Befehl aus Listing 22.28 verwenden:

```
root@saturn:~# ip route flush cache
```

Listing 22.28 Löschen des Routing-Cache

22.3.2 Da geht noch mehr: »Advanced Routing«

Beim klassischen Routing wird anhand der Zieladresse ermittelt, welche Route genutzt werden muss. Dabei hat stets die Route Vorrang, die das Ziel am genauesten beschreibt – also die Route mit der größten Netzwerkmaske.

Dadurch sind die Möglichkeiten, eine Routing-Entscheidung zu treffen, natürlich limitiert. Beim Advanced Routing werden diese Möglichkeiten stark erweitert. Dabei ist es nämlich möglich, die Routing-Entscheidung anhand weiterer Kriterien zu treffen, beispielsweise anhand der Quelladresse, des eingehenden Interface oder auch, unter Mithilfe von iptables, anhand von Ports und Protokollen. Diese Routing-Art wird auch als *Policy Based Routing* oder kurz *Policy Routing* bezeichnet. Das Policy Routing wird durch die Verwendung mehrerer Routing-Tabellen und der sogenannten *Policy Routing Database* (PRDB) ermöglicht. Die PRDB wird mit dem `ip rule`-Kommando verändert. In ihr stehen Regeln, die abgearbeitet werden, um die passende Routing-Tabelle herauszufinden. Streng genommen wird immer

mit Policy Routing gearbeitet, auch wenn Sie dies bei den Tools `route` oder `ip route` nicht gesondert angeben. Standardmäßig sind bereits einige Tabellen und Regeln vorhanden, über die das klassische Routing gebildet wird.

22.3.3 Die vorhandenen Regeln ansehen

In Listing 22.29 sehen Sie ein Beispiel dafür, wie Sie die vorhandenen Regeln ausgeben lassen können. Dort sind bereits drei Regeln vorhanden. Die erste Spalte gibt die Priorität der Regel an. Dabei gilt: je kleiner der Wert, desto höher die Priorität. Daher wird immer die Regel mit dem niedrigsten Wert zuerst ausgeführt – also die Regel mit der Priorität 0. Diese Regel kann nicht geändert oder gelöscht werden.

```
root@saturn:~# ip rule show
0:      from all lookup local
32766:  from all lookup main
32767:  from all lookup default
```

Listing 22.29 Vorhandene Regeln ausgeben

In der nächsten Spalte wird mit `from all` angegeben, dass diese Regel für alle Pakete gilt. Das Schlagwort `lookup` gibt an, in welcher Routing-Tabelle nachgeschlagen werden soll. Dieser Standardregelsatz sorgt also dafür, dass nacheinander die Routing-Tabellen `local`, `main` und `default` nach passenden Routing-Einträgen durchsucht werden. Bei einem Treffer wird die Verarbeitung beendet.

Die Routing-Tabelle `local` wird vom Kernel befüllt. Dort sind Informationen über lokale und Broadcast-Routen enthalten. Die Routing-Tabelle `main` enthält die Routen, die mit den Tools `route` oder `ip route` gesetzt wurden. Die letzte Routing-Tabelle, `default`, ist standardmäßig leer. Sie können auch gezielt den Inhalt einer Tabelle mit dem Befehl `ip route show table <TABELNAME>` ausgeben lassen, so wie es in Listing 22.30 dargestellt ist:

```
root@saturn:~# ip route show table local
broadcast 127.0.0.0 dev lo proto kernel scope link src 127.0.0.1
local 127.0.0.0/8 dev lo proto kernel scope host src 127.0.0.1
local 127.0.0.1 dev lo proto kernel scope host src 127.0.0.1
broadcast 127.255.255.255 dev lo proto kernel scope link src 127.0.0.1
broadcast 192.168.0.0 dev eth0 proto kernel scope link src 192.168.0.163
local 192.168.0.163 dev eth0 proto kernel scope host src 192.168.0.163
broadcast 192.168.0.255 dev eth0 proto kernel scope link src 192.168.0.163
```

Listing 22.30 Auflisten der Routing-Tabellen für das lokale Netz

Listing 22.31 zeigt die Ausgabe der Standardtabelle `main`. Diese Ausgabe sollte Ihnen bekannt vorkommen.

```
root@saturn:~# ip route show table main
default via 192.168.0.1 dev eth0
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.163
```

Listing 22.31 Ausgabe der Standard-Routing-Tabelle

Der Vollständigkeit halber haben wir in Listing 22.32 die Routing-Tabelle default ausgeben lassen – wie erwartet hat diese keinen Inhalt:

```
root@saturn:~# ip route show table default
```

Listing 22.32 Die »default«-Tabelle ist leer.

22.3.4 Eine neue Routing-Tabelle anlegen

Eigentlich müssen Routing-Tabellen nicht angelegt werden, da dies bei der Benutzung automatisch geschieht. Ein Linux-System kann maximal mit 256 Routing-Tabellen umgehen, da jede Tabelle über eine Nummer zwischen 0 und 255 identifiziert wird.

Deutlich angenehmer und für den Benutzer verständlicher ist das Ansprechen einer Tabelle über Namen. Die Tabellennamen werden in der Datei */etc/iproute2/rt_tables* definiert – wie zu erwarten ist, sind dort auch die Standardtabellen definiert (siehe Listing 22.33):

```
root@saturn:~# cat /etc/iproute2/rt_tables
#
# reserved values
#
255    local
254    main
253    default
0      unspec
```

Listing 22.33 Liste der Tabellennamen

22.3.5 Ändern der Policy Routing Database

Zum Erstellen und Löschen von Regeln werden die Befehle `ip rule add` und `ip rule delete` verwendet. Regeln bestehen aus drei Werten: aus den Kriterien für die Regel, aus der Tabelle, die angesprochen wird, und aus der Priorität für die Regel.

Diese Angaben sind allerdings optional und können somit auch weggelassen werden – was selbstverständlich wenig sinnvoll ist, aber die Funktionsweise verdeutlicht:

```
root@saturn:~# ip rule show
0:      from all lookup local
32766:  from all lookup main
32767:  from all lookup default
```

```
root@saturn:~# ip rule add
root@saturn:~# ip rule show
0:      from all lookup local
32765:  from all lookup main
32766:  from all lookup main
32767:  from all lookup default
```

Listing 22.34 Beispiel für »ip rule show«

Wie Sie Listing 22.34 entnehmen können, wurde die neue Routing-Tabelle erzeugt. Dabei wurde als Standardkriterium `from all` verwendet, die Zieltabelle auf `main` und die Priorität auf 32765 gesetzt. Die Priorität errechnet `ip` dabei nach einer festen Formel: *niedrigste bereits vorhandene Zahl* – 1. Wird zum Beispiel eine Regel mit der Priorität 5000 erstellt, so wird der nächsten Regel automatisch die Priorität 4999 zugewiesen. Die wichtigsten Bedingungen zur Definition einer Regel sind:

- ▶ `from <NETWORK>`
spezifiziert das Quellnetz, für das die Regel gelten soll.
- ▶ `to <NETWORK>`
spezifiziert das Zielnetz.
- ▶ `iif <INTERFACE>`
gibt das eingehende Interface an.
- ▶ `fwmark <MARK>`
Auswahl von Paketen, die von *iptables* markiert wurden



Um beispielsweise für alle Pakete, die über `eth1` eingehen, eine andere Routing-Tabelle anzuspringen, müssen Sie folgende Befehle absetzen:

```
root@saturn:~# echo 100 table_in_eth0 >> /etc/iproute2/rt_tables
root@saturn:~# ip rule add iif eth1 table table_in_eth0 priority 10000
root@saturn:~# ip rule show
0:      from all lookup local
10000:  from all iif eth1 lookup table_in_eth0
32765:  from all lookup main
32766:  from all lookup main
32767:  from all lookup default
```

Listing 22.35 Nutzung einer anderen Routing-Tabelle für Pakete auf »eth1«

Um Regeln zu löschen, können Sie sich einfach so auf deren Priorität beziehen, wie in Listing 22.36 dargestellt:

```
root@saturn:~# ip rule delete priority 10000
```

Listing 22.36 Löschen einer Regel mit Angabe der Priorität



Achten Sie auf die Prioritäten!

Geben Sie immer eine Priorität mit an, damit Sie die Kontrolle über die Reihenfolge der Regeln haben. Sie müssen zusätzlich darauf achten, dass Sie keine Priorität doppelt vergeben. Dies wird leider nicht von `ip` abgefangen. Kontrollieren Sie daher nach Veränderungen immer den gegenwärtigen Zustand mittels `ip rule show!`

22

Falls Sie über mehrere Regeln mit der gleichen Priorität verfügen, müssen Sie den Befehl zum Löschen entweder so oft absetzen, bis alle Regeln dieser Priorität entfernt sind (pro Befehl wird nämlich immer nur der erste Treffer entfernt), oder Sie geben die zu löschende Regel vollständig an, wie in Listing 22.37 dargestellt:

```
root@saturn:~# ip rule delete from all iif eth1 table table_in_eth0
```

Listing 22.37 Löschen einer Regel bei doppelter Priorität

22.3.6 Routing über mehrere Uplinks

Falls Sie über mehrere Internetanbindungen verfügen, müssen Sie festlegen, über welchen Weg Ihre Clients ins Internet geroutet werden sollen. Nehmen wir an, Sie verfügen über zwei lokale Netze, die über zwei verschiedene DSL-Router ins Internet gelangen sollen (siehe Abbildung 22.1). Dies ist relativ einfach zu lösen.

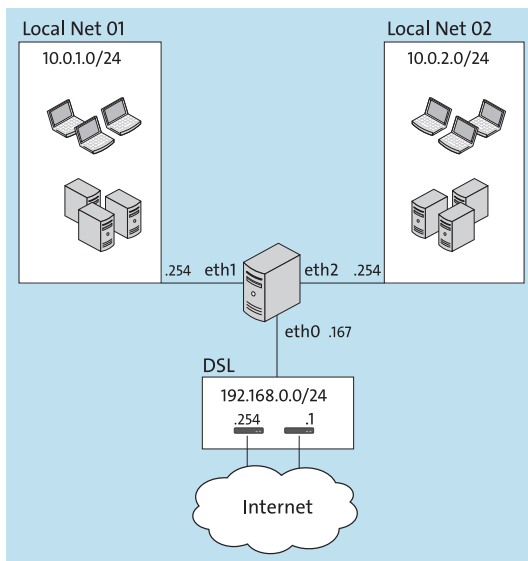


Abbildung 22.1 Einfach zu lösen: zwei lokale Netze am DSL-Router

Wie Sie Abbildung 22.1 entnehmen können, befinden sich sowohl der Linux-Router als auch



die DSL-Router im Netzwerk 192.168.0.0/24. Die DSL-Router verwenden die IP-Adressen 192.168.0.1 und 192.168.0.254. Die lokalen Netzwerke sind an zwei Interfaces an den Linux-Router angebunden: das *Local Net 01* an eth1 mit dem Netzwerk 10.0.1.0/24 und das *Local Net 02* an eth2 mit dem Netzwerk 10.0.2.0/24. Das Standardgateway ist die 192.168.0.1.

Um den Anforderungen zu entsprechen, müssen Sie zunächst eine neue Routing-Tabelle anlegen bzw. deren Namen definieren. Anschließend müssen Sie Regeln erstellen, damit diese Routing-Tabelle angesprochen wird. Führen Sie zum Anlegen der Routing-Tabelle die Befehle aus Listing 22.38 aus:

```
root@saturn:~# echo "100 provider_B" >> /etc/iproute2/rt_tables
root@saturn:~# ip rule add from 10.0.2.0/24 table provider_B prio 10000
root@saturn:~# ip rule show
0:      from all lookup local
10000:  from 10.0.2.0/24 lookup provider_B
32765:  from all lookup main
32766:  from all lookup main
32767:  from all lookup default
```

Listing 22.38 Erstellen der Routing-Tabelle »provider_B«

Wie Sie Listing 22.38 entnehmen können, wurde die Tabelle `provider_B` mit einer Priorität von 10.000 angelegt und so konfiguriert, dass sie auf alle Pakete aus dem Netz 10.0.2.0/24 zutrifft. Damit die Pakete aus dem Netz 10.0.2.0/24 nun auch zum Provider B geroutet werden, müssen Sie noch die entsprechende Route in der Tabelle erzeugen:

```
root@saturn:~# ip route add default via 192.168.0.254 table provider_B
```

Listing 22.39 Füllen der Routing-Tabelle »provider_B«

Wie Sie Listing 22.39 entnehmen können, wurde die Routing-Tabelle mit dem Schlagwort `table` übergeben. Nun werden alle Anfragen aus dem *Local Net 02* über den Provider B geroutet. Die Pakete aus *Local Net 01* hingegen werden weiterhin über Provider A geleitet, da dort das Standardgateway aus der Tabelle `main` greift. Die entsprechenden Routing-Tabellen haben wir in Listing 22.40 aufgelistet:

```
root@saturn:~# ip route show table main
10.0.1.0/24 dev eth1 proto kernel scope link src 10.0.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.163 metric 1
10.0.2.0/24 dev eth2 proto kernel scope link src 10.0.2.1
default via 192.168.0.1 dev eth0 proto static
root@saturn:~# ip route show table provider_B
default via 192.168.0.254 dev eth0
```

Listing 22.40 Anzeige der Tabellen »main« und »provider_B«

Damit Ihr Linux-Router die Pakete auch verarbeiten kann, müssen Sie das Routing aktivieren. Falls dies noch nicht der Fall ist, führen Sie den Befehl aus Listing 22.41 aus:

```
root@saturn:~# echo "1" > /proc/sys/net/ipv4/ip_forward
```

Listing 22.41 Routing aktivieren

22

Damit ist die Konfiguration komplett. Wie Sie sehen, ist diese Anforderung mit `iproute2` sehr einfach umsetzbar.

Steigern wir den Schwierigkeitsgrad, indem wir das bestehende Setup erweitern (siehe Abbildung 22.2). Über einen VPN-Tunnel wird ein Rechenzentrum mit dem Netz `172.16.100.0/24` angebunden. Die externe IP-Adresse des Routers und des VPN-Servers im Rechenzentrum ist die `172.16.100.1`. Unser lokaler Endpunkt des VPN hat die IP-Adresse `10.99.99.254`, das System im Rechenzentrum die `10.99.99.1`. Die Administratoren aus dem Netzwerk *Local Net 01* sollen die Systeme im Rechenzentrum (`172.16.100.0/24`) via `ssh` (`tcp/22`) über den VPN-Tunnel erreichen können und nicht über das Internet. Die Serverdienste `pop3` (`tcp/110`), `imap` (`tcp/143`) und `smtp` (`tcp/25`) des Mailservers mit der IP-Adresse `172.16.100.200` sollen ebenfalls über den VPN-Tunnel laufen. Der sonstige Verkehr in das Rechenzentrum soll über den ersten DSL-Router (`192.168.0.1` – Provider A) geleitet werden. Die bisher eingerichtete Verteilung des Internetzugriffs soll bestehen bleiben.

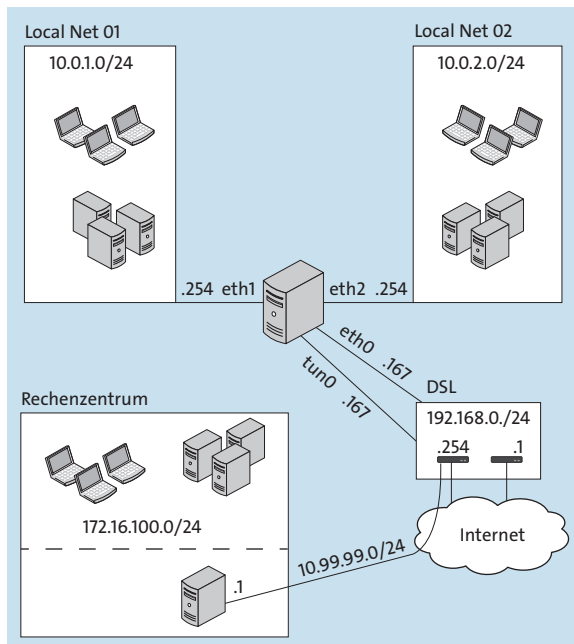


Abbildung 22.2 Fortgeschritten: Einbindung einer VPN-Verbindung



Nur ein Ansatz

Die folgende Lösung ist nur eine von mehreren möglichen. Sie ist weder die simpelste noch die eleganteste, dafür aber eine leicht nachvollziehbare Lösung.

Um diese Anforderungen abbilden zu können, müssen Sie zunächst zwei weitere Routing-Tabellen anlegen: eine für das Rechenzentrum und eine für die Verbindung über Provider A. Die Letztere ist zwar nicht zwingend erforderlich, erhöht aber die Übersichtlichkeit. In Listing 22.42 sehen Sie die dafür notwendigen Befehle:

```
root@saturn:~# echo "200 provider_A" >> /etc/iproute2/rt_tables
root@saturn:~# echo "50 datacenter" >> /etc/iproute2/rt_tables
```

Listing 22.42 Erstellen der zusätzlichen Routing-Tabellen

Anschließend setzen wir eine Regel für Provider A und die dazu passende Route, damit der Datenverkehr vom *Local Net 01* (10.0.1.0/24) über den Provider in Richtung Internet geleitet wird. Danach setzen wir eine Regel und die dazu passende Route, um unseren Verkehr von 10.0.1.0/24 über das erste DSL-Modem zu senden (siehe Listing 22.43):

```
root@saturn:~# ip rule add from 10.0.1.0/24 table provider_A prio 10001
root@saturn:~# ip route add default via 192.168.0.1 table provider_A
```

Listing 22.43 DSL-Modem Routen für »Local Net 01« über »Provider A« setzen

Bisher haben wir das gleiche Ergebnis wie aus dem ersten Beispiel (siehe Listing 22.40), nur mit einer separaten Tabelle für Provider A. Nun definieren wir eine Regel, die den gesamten Traffic in das Rechenzentrum über Provider A leitet. Da diese Regel quellnetzunabhängig sein soll, bekommt sie einen niedrigeren Wert, also eine höhere Priorität (siehe Listing 22.44):

```
root@saturn:~# ip rule add to 172.16.100.0/24 table provider_A prio 9000
root@saturn:~# ip rule show
0:      from all lookup local
9000:   from all to 172.16.100.0/24 lookup provider_A
10000:  from 10.0.2.0/24 lookup provider_B
10001:  from 10.0.1.0/24 lookup provider_A
32766:  from all lookup main
32767:  from all lookup default
```

Listing 22.44 Regel für den Datenverkehr zum Rechenzentrum über Provider A

Für die Umsetzung der Portfreigaben greifen wir auf iptables zurück. Mit iptables werden die Pakete markiert, die die Bedingungen für den VPN-Tunnel erfüllen. Diese Markierungen kann ip wiederum auslesen und den Datenverkehr in die jeweilige Routing-Tabelle leiten (siehe Listing 22.45):

```

root@saturn:~# iptables -A PREROUTING -t mangle -p tcp -m multiport \
--dport 25,110,143 -d 172.16.100.200 -j MARK --set-mark 1
root@saturn:~# iptables -A PREROUTING -t mangle -p tcp --dport 22 \
-s 10.0.1.0/25 -d 172.16.100.0/24 -j MARK --set-mark 1

```

Listing 22.45 Markieren der Pakete mit »iptables«

22

Neugierig geworden?

Mehr zum Thema *iptables* finden Sie in Abschnitt 22.6, »Firewalls mit »netfilter« und »iptables««.



Mit den Regeln aus Listing 22.45 werden diejenigen Pakete markiert, die auf die Bedingungen aus dem Beispiel zutreffen. Ein Teil der Aufgabe (die Identifikation anhand von Quell- und Zieladresse) hätten Sie auch mit *ip* lösen können – dann wäre lediglich der portbasierte Teil auf *iptables* zurückgefallen. Diese Vorgehensweise ist aber nicht ratsam, da Sie mit ihr unterschiedliche Mechanismen für eine Aufgabe verwenden und somit die Übersichtlichkeit nicht gerade erhöhen!

Abschließend muss nur noch definiert werden, wie *ip* mit den markierten Paketen umgehen soll. Da auch diese Regel vor den bisherigen verarbeitet werden muss, geben wir auch dieser Regel einen niedrigeren Wert. Zusätzlich muss der Routing-Eintrag für den Datenverkehr über den VPN-Tunnel (auf das Interface *tun0*) erstellt werden. In Listing 22.46 sind die entsprechenden Befehle aufgeführt:

```

root@saturn:~# ip rule add fwmark 1 table datacenter prio 5000
root@saturn:~# ip route add default via 10.99.99.254 dev tun0 table datacenter

```

Listing 22.46 Regel für markierte Pakete und Erstellen einer Route für den VPN-Tunnel

Unterschiedliche Darstellung der Markierungsnummern

Die Markierungsnummer ist ein hexadezimaler Wert. Sie sollten sich daher nicht wundern, wenn aus der Markierung 10 in der Anzeige *0xa* wird.



Sehen wir uns nun in Listing 22.47 das vollständige Regelwerk noch einmal an:

```

root@saturn:~# ip rule show
0:      from all lookup local
5000:   from all fwmark 0x1 lookup datacenter
9000:   from all to 172.16.100.0/24 lookup provider_A
10000:  from 10.0.2.0/24 lookup provider_B
10001:  from 10.0.1.0/24 lookup provider_A

```

```
32766: from all lookup main
32767: from all lookup default
```

```
root@saturn:~# ip route show table main
172.16.30.1 via 192.168.0.1 dev eth0 proto static
10.99.99.254 dev tun0 proto kernel scope link src 10.99.99.1
10.0.1.0/24 dev eth1 proto kernel scope link src 10.0.1.1
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.163 metric 1
10.0.2.0/24 dev eth2 proto kernel scope link src 10.0.2.1
default via 192.168.0.1 dev eth0 proto static
```

```
root@saturn:~# ip route show table provider_A
default via 192.168.0.1 dev eth0
```

```
root@saturn:~# ip route show table provider_B
default via 192.168.0.254 dev eth0
```

```
root@saturn:~# ip route show table datacenter
default via 10.99.99.254 dev tun0
```

Listing 22.47 Anzeige der kompletten Konfiguration

22.3.7 Fazit bis hierher

Das Tool `ip` bietet Ihnen unzählige Möglichkeiten, um auch die komplexesten Setups abbilden zu können. Insbesondere im Zusammenspiel mit `iptables` wird es geradezu magisch. Das Tool kann aber noch viel mehr. Die *Manpage* beinhaltet viele nützliche Tipps und Tricks. Wagen Sie einen Blick hinein, es wird sich lohnen! Die Übersicht zu behalten ist das A und O, gerade wenn es um die Fehlersuche geht. Beherzigen Sie daher unsere Hinweise, sprechende Namen zu verwenden und Konfigurationen bzw. Mechanismen nicht zu verteilen, sondern diese gebündelt vorzunehmen.

22.4 Bonding

Zur Erhöhung der Ausfallsicherheit oder des Datendurchsatzes können mehrere physische Netzwerkinterfaces zu einem logischen zusammengeschlossen werden. Linux-Admins nennen diesen Vorgang *Bonding*. Bei den Herstellern von Netzwerkhardware werden oft andere Begriffe dafür verwendet – leider überschneiden sich diese auch, sodass das Chaos vorprogrammiert ist. Besonders *Cisco* weicht dabei von der Norm ab. Tabelle 22.1 zeigt Ihnen, was wirklich gemeint ist, damit Sie mit dem Netzwerkmitarbeiter Ihres Vertrauens auch eine gemeinsame Sprache finden.

Hersteller	Trunking	Channeling
Cisco	Trunk	Etherchannel
HP	LACP/Link Aggregation	Trunk
Sun	VLANs	Trunk
Linux	802.1q	Bonding

Tabelle 22.1 Begriffe und deren Bedeutung

Spricht also ein Cisco-Administrator von einem *Etherchannel* oder ein HP-Administrator von einem *Trunk*, so müssen Sie als Linux-Administrator das *Bonding* konfigurieren. Zusätzlich wird (gerade bei Windows-Administratoren) auch gerne von *NIC-Teaming* gesprochen.

22.4.1 Bonding-Konfiguration

Damit Sie auf Ihrem System ein Bond konfigurieren können, müssen Sie zunächst sicherstellen, dass der dafür benötigte Treiber geladen ist und mit dem richtigen Parameter gestartet wurde. Führen Sie dafür den Befehl aus Listing 22.48 aus:

```
root@saturn:~# modprobe bonding mode=active-backup
```

Listing 22.48 Laden des Bonding-Treibers

Anschließend benötigen Sie das Paket *ifenslave*, damit Sie ein Bond über das gleichnamige Kommando konfigurieren können. Das Paket ist bei allen Linux-Distributionen Bestandteil der Paketquellen, installieren Sie es daher über den gewohnten Weg. Um ein Bond aus den Interfaces *eth0* und *eth1* zu erzeugen, müssen Sie die Befehle aus Listing 22.49 absetzen:

```
root@saturn:~# ip link add bond0 type bond
root@saturn:~# ip link set up bond0
root@saturn:~# ifenslave bond0 eth1 eth2
```

Listing 22.49 Hinzufügen der Netzwerkkarten zum Bonding-Interface

Wie Sie Listing 22.49 entnehmen können, wurde das neue Gerät *bond0* erzeugt. Dieses finden wir sogleich in der Linkliste, die in Listing 22.50 gezeigt wird:

```
root@saturn:~# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state\
    UNKNOWN qlen 1000
    link/ether 00:26:18:81:1d:63 brd ff:ff:ff:ff:ff:ff
```

```
3: eth1: <BROADCAST,NOARP,SLAVE,UP,LOWER_UP> mtu 1500 qdisc noqueue\
    master bond0 state UNKNOWN
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
4: eth2: <BROADCAST,NOARP,SLAVE,UP,LOWER_UP> mtu 1500 qdisc noqueue\
    master bond0 state UNKNOWN
    link/ether 08:00:27:dd:9c:fd brd ff:ff:ff:ff:ff:ff
7: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc\
    noqueue state UP
    link/ether 08:00:27:05:6d:c4 brd ff:ff:ff:ff:ff:ff
```

Listing 22.50 Auflisten aller Interfaces

Um nähere Informationen zu dem soeben erstellten Gerät `bond0` zu erhalten, können Sie einen Blick in das `proc`-Dateisystem wagen. Unter `/proc/net/bonding/bond0` finden Sie das Gerät, das wir uns nun genauer ansehen (siehe Listing 22.51):

```
root@saturn:~# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v5.10.0-20-amd64
```

```
Bonding Mode: fault-tolerance (active-backup)
Primary Slave: None
Currently Active Slave: eth1
MII Status: up
MII Polling Interval (ms): 0
Up Delay (ms): 0
Down Delay (ms): 0
```

```
Slave Interface: eth1
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:05:6d:c4
Slave queue ID: 0
```

```
Slave Interface: eth2
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:dd:9c:fd
Slave queue ID: 0
```

Listing 22.51 Informationen zum Bonding-Interface

Wie Sie in Listing 22.51 sehen, werden nicht nur die Teilnehmer des Bonds dargestellt, sondern es wird auch gezeigt, welcher zurzeit aktiv ist. Darüber hinaus können Sie sehen, welcher *Modus* verwendet wird (in Listing 22.51 ist es *active-backup*). Wenn Sie mit dem Ergebnis zufrieden sind, können Sie dem neuen Interface per `ip address add` eine IP-Adresse geben und es benutzen. Neben dem gezeigten *active-backup*-Modus gibt es noch weitere Konfigurationsmöglichkeiten:

- ▶ **Modus 0: »balancer-rr« (Lastverteilung und Ausfallsicherheit)**
Das *rr* im Namen steht für *Round Robin*. Bei diesem Verfahren werden die abgesandten Pakete auf die gebündelten Interfaces verteilt. Eingehend kann nur ein Interface genutzt werden. Dieser Modus führt also zu einer Erhöhung des möglichen Uploads und erhöht gleichzeitig die Ausfallsicherheit.
- ▶ **Modus 1: »active-backup« (Ausfallsicherheit)**
Hierbei ist nur ein Interface aktiv. Das passive Interface wird nur aktiviert, wenn das vormals aktive ausfällt. Diese Methode wird zum *Failover* eingesetzt.
- ▶ **Modus 2: »balance-xor« (Lastverteilung und Ausfallsicherheit)**
Das *xor* im Namen steht für die Berechnungsmethode *XOR*. Bei ihr wird jedem Ziel im Netzwerk ein Interface fest zugewiesen. Die Zuordnung erfolgt dabei über den Modulo der Division zwischen $\langle \text{Quell-MAC} \rangle \text{ XOR } \langle \text{Ziel-MAC} \rangle$ und der Anzahl der Slave-Interfaces.
- ▶ **Modus 3: »broadcast« (Lastverteilung und Ausfallsicherheit)**
Hier werden die Daten auf allen Interfaces geflutet. Dies führt zu stark erhöhtem Netzwerkverkehr. Setzen Sie diese Methode nur ein, wenn es zwingend erforderlich ist.
- ▶ **Modus 4: »802.3ad/LACP« (Lastverteilung und Ausfallsicherheit)**
Der IEEE-Standard 802.3ad, auch als *LACP*¹ oder einfach als *Link Aggregation* bezeichnet, ermöglicht es, die Interfaces gemeinsam zu nutzen. Damit kann die maximal mögliche Bandbreite erhöht werden. Dabei gibt es Einschränkungen: Alle Schnittstellen müssen über dieselben *Speed*- und *Duplex*-Einstellungen verfügen und an ein und denselben Switch angeschlossen sein.
- ▶ **Modus 5: »balance-tlb« (Lastverteilung)**
Das *tlb* im Namen steht für *transmit load balancing*. Hier wird, wie beim *balance-xor*, jedem Ziel im Netzwerk ein Interface fest zugewiesen. Die Zuordnung erfolgt dabei über eine deutlich komplexere und effizientere Berechnungsmethode.
- ▶ **Modus 6: »balance-alb« (Lastverteilung)**
Das *alb* im Namen steht für *adaptive load balancing*. Es handelt sich um eine Erweiterung des *balance-tlb*, wobei eingehende Verbindungen ebenfalls auf unterschiedliche Interfaces verteilt werden.

¹ *Link Aggregation Control Protocol*, engl. für *Schnittstellenbündelungskontrollprotokoll*

Den Datendurchsatz mit einem Bond zu erhöhen, das klingt zunächst vielversprechend. Unter optimalen Bedingungen (die sich meist nur im Labor erzeugen lassen) können mit vier 1-Gbit-Interfaces durchaus 4 Gbit/s erreicht werden. Allerdings sind diese Werte in der Praxis stark vom gewählten Bonding-Modus und von den lokalen Gegebenheiten abhängig. Heutzutage wird in den meisten Fällen das Bonding eher zur Erhöhung der Verfügbarkeit verwendet.

Zusätzlich zum Bonding-Modus können Sie beim Laden des Moduls auch weitere Parameter angeben. Diese werden umfangreich in der Manpage beschrieben.

22.4.2 Bonding unter Debian

Bisher haben wir das Bond lediglich zur Laufzeit konfiguriert. Nach jedem Neustart des Systems müssten Sie diese Schritte also stets wiederholen. In diesem Abschnitt zeigen wir Ihnen, wie Sie die Konfiguration bootresistent einrichten können.

Die Netzwerkkonfiguration wird in `/etc/network/interfaces` durchgeführt. Damit das Bonding direkt beim Systemstart angelegt wird, müssen Sie die Zeilen aus Listing 22.52 der bestehenden Konfiguration hinzufügen:

```
auto bond0
iface bond0 inet static
    address 172.16.1.1
    netmask 255.255.255.0
    slaves eth1 eth2
    bond-mode active-backup
    bond-miimon 100
    bond-downdelay 200
    bond-updelay 200
```

Listing 22.52 Auszug aus der Datei »`interfaces`«

Mit dieser Konfiguration werden die Interfaces `eth1` und `eth2` zum `bond0` zusammengeschlossen. Dabei wird die Betriebsart auf `active-backup` gesetzt und das Interface `eth1` als initial aktives Interface bestimmt. Weitere Informationen zu Einrichtungen finden Sie in der Manpage zu `interfaces` und in der Dokumentation unter `/usr/share/doc/ifenslave`.

22.4.3 Bonding unter Ubuntu

Seit Ubuntu 18.04 wird die Netzwerkkonfiguration von `netplan` übernommen. Für ein Bond müssen Sie in der entsprechenden YAML-Konfigurationsdatei unterhalb von `/etc/netplan` die Konfiguration anpassen. Listing 22.53 zeigt, wie Sie die Interfaces `enp0s8` und `enp0s9` in ein Active-Backup-Bond schalten:

```

network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: yes
    enp0s8:
      match:
        name: enp0s8
    enp0s9:
      match:
        macaddress: 08:00:27:a2:23:60
  bonds:
    bond0:
      interfaces: [enp0s8, enp0s9]
      addresses: [10.0.0.1/24]
      parameters:
        mode: active-backup
        mii-monitor-interval: 1
        primary: enp0s8

```

Listing 22.53 »netplan«-Konfiguration unter Ubuntu

22.4.4 Bonding unter CentOS

Bei CentOS-Systemen wird die Netzwerkkonfiguration in Dateien namens *ifcfg-<INTERFACE-NAME>* unterhalb von */etc/sysconfig/network-scripts/* vorgenommen. Prinzipiell können Sie diese Dateien auch von Hand erstellen. Allerdings bringt CentOS von Haus aus das Tool *nmcli* zur Netzwerkkonfiguration mit. In Listing 22.54 sehen Sie die notwendigen Befehle, um ein Bond mit zwei Slaves anzulegen und dieses im Active-Backup-Modus zu betreiben:

```

### Bond anlegen:
[root@centos ~]# nmcli con add type bond con-name bond0 ifname bond0 mode \
active-backup ip4 <IP> gw4 <GW>

### Slave hinzufügen:
[root@centos ~]# nmcli con add type bond-slave ifname enp0s8 master bond0
[root@centos ~]# nmcli con add type bond-slave ifname enp0s9 master bond0

### Konfiguration aktivieren:
[root@centos ~]# nmcli con down bond0 ; nmcli con up bond0
[root@centos ~]# systemctl restart NetworkManager

```

Listing 22.54 Erstellen eines Bonds mit zwei Slaves unter CentOS

22.4.5 Bonding unter openSUSE Leap

Unter openSUSE Leap wird die Netzwerkkonfiguration in Dateien mit der Bezeichnung *ifcfg- \langle INTERFACENAME \rangle* unterhalb von */etc/sysconfig/network/* vorgenommen. Die genaue Syntax finden Sie in der Manpage von *ifcfg* und *ifcfg-bonding*. Für das Äquivalent des bisher erzeugten Bonds müssen Sie die Datei *ifcfg-bond0* mit dem Inhalt aus Listing 22.55 anlegen:

```
### /etc/sysconfig/network/ifcfg-bond0
STARTMODE='auto'
BOOTPROTO='static'
IPADDR='172.16.2.2/24'
BONDING_MASTER='yes'
BONDING_SLAVE_0='eth1'
BONDING_SLAVE_1='eth2'
BONDING_MODULE_OPTS='mode=active-backup miimon=100'

### /etc/sysconfig/network/ifcfg-eth1
STARTMODE='hotplug'
BOOTPROTO='none'

### /etc/sysconfig/network/ifcfg-eth2
STARTMODE='hotplug'
BOOTPROTO='none'
```

Listing 22.55 Inhalt der Konfigurationsdateien »ifcfg-bond0«, »ifcfg-eth1« und »ifcfg-eth2«

22.5 IPv6

IPv6 ist das offizielle Nachfolgeprotokoll von IPv4. Irgendwann soll es IPv4 gänzlich ablösen. Der Grund dafür ist recht simpel: IP-Adressknappheit. IPv4-Adressen haben eine Länge von 32 Bit, das heißt, theoretisch stehen 2^{32} Adressen zur Verfügung. Von diesen ca. 4 Milliarden Adressen stehen aber nicht alle zur Nutzung bereit. Von den 256 möglichen /8-Netzen sind allein 35 Netze für spezielle Nutzungszwecke reserviert, beispielsweise 127.0.0.0/8 für Loopback-Adressen, 10.0.0.0/8 für private Nutzung oder 224.0.0.0/8 für Multicast-Adressen (siehe Tabelle 22.2).

IP-Bereich	Verwendung
10.0.0.0/8	private Netzwerkadresse RFC1918
127.0.0.0/8	Loopback (localhost)

Tabelle 22.2 Liste der reservierten IPv4-Bereiche

IP-Bereich	Verwendung
169.254.0.0/16	link local (APIPA)
172.16.0.0/12	private Netzwerkadresse RFC1918
192.0.2.0/24	Test und Dokumentation
192.168.0.0/16	private Netzwerkadresse RFC1918
224.0.0.0/4	Multicast-Adressen
240.0.0.0/4	experimentelle Adressen

Tabelle 22.2 Liste der reservierten IPv4-Bereiche (Forts.)

Zu allem Übel fehlen darüber hinaus noch mal 40 /8-Netze, die in den Anfangstagen, als der Erfolg des Internets noch nicht absehbar war, großzügig an einzelne Firmen oder Organisationen vergeben wurden. Von den verbleibenden 181 /8-Netzen sind aktuell nur noch wenige nicht vergeben – es wird geschätzt, dass mehr als 95 % der Adressen verbraucht sind. Die Adressen werden von der *Internet Assigned Numbers Authority* (IANA) verwaltet und an *Regional Internet Registries* (RIR) vergeben. Davon gibt es fünf weltweit. Das *RIPE NCC*² ist beispielsweise die RIR für Europa, Zentralasien und den Nahen Osten.

Die RIRs wiederum verteilen IP-Adressen an *Local Internet Registries* (LIR). Erst von den LIRs werden IP-Adressen dann an Endkunden vergeben.

Zwischenzeitlich wurde damit gerechnet, dass die IP-Adressen ausgehen würden. So rechnete das RIPE NCC vor³, dass die letzten Adressen für Europa im August 2012 vergeben würden. Wie wir alle wissen, ist dies nicht eingetroffen.

Die ergriffenen Maßnahmen haben die Situation etwas entspannt: Zum einen wurde eine Umverteilung der vergebenen Netze vorgenommen, und zum anderen haben viele Provider begonnen, die IPv6-Einführung deutlich zu beschleunigen. Daher sind immer noch (wenn auch sehr wenige) IPv4-Adressen verfügbar.

Es gibt also keinen Grund, in Panik zu verfallen. Aber Sie sollten sich langsam mit dem Gedanken anfreunden, dass IPv6 auch zu Ihnen kommen wird und Sie Ihr Netzwerk dementsprechend anpassen müssen.

Durch die Verwendung von NAT auch aufseiten der Provider wird die Knappheit etwas gemildert, aber da die IPv4-Adressen auch immer teurer werden, lohnt sich die Umstellung schon allein aus Kostengründen ab einem bestimmten Zeitpunkt.

² Réseaux IP Européens Network Coordination Centre

³ Siehe www.potaroo.net/tools/ipv4/index.html

22.5.1 Die Vorteile von IPv6

Einer der Hauptvorteile von IPv6 ist die Vergrößerung des IP-Adressraums: Der Adressraum bei IPv6 beträgt 128 Bit. Die minimale Netzgröße, die an Endkunden herausgegeben werden soll, ist ein /64-Netz. Ein einzelner Endkunde bekommt also mehr eigene IP-Adressen, als in IPv4 insgesamt zur Verfügung stehen. Damit ist dann auch wieder das Ende-zu-Ende-Prinzip erfüllt, das heißt, jeder Host ist im Internet garantiert direkt ansprechbar. Zurzeit ist das aufgrund der IP-Adressknappheit leider nicht möglich. Ohne *Network Address Translation* (NAT) wäre es jetzt schon undenkbar, mit mehr als einem Gerät von zu Hause aus in das Internet zu gehen. NAT ist aber nur eine Notlösung, und einige Protokolle können nicht vernünftig mit NAT arbeiten, sodass es immer wieder zu Problemen kommt.

Was heißt das eigentlich in Zahlen? Wie viele IPv6-Adressen gibt es denn nun? Deshalb hier einmal ein kleines Zahlenspiel: IPv6-Adressen bestehen aus 128 Bit; das sind dann also 2^{128} mögliche Hostadressen. Als Zahl sieht das so aus: $2^{128} = 340.282.366.920.938.463.463.374.607.431.768.211.456$ mögliche Adressen. Das entspricht ungefähr 665 Milliarden IPv6-Adressen pro Quadratmillimeter der Erdoberfläche: Da ist auch bestimmt für Ihre Kaffeemaschine im Büro noch eine IP-Adresse frei. Natürlich gibt es auch bei IPv6 wieder reservierte Bereiche, sodass die Zahl etwas kleiner wird, aber die Menge sollte immer noch ausreichend für zukünftige Techniken sein.

22.5.2 Notation von IPv6-Adressen

Der gigantische Adressraum von IPv6 hat leider auch Nachteile. Mit einer Länge von 128 Bit sind IPv6-Adressen nicht gerade einprägsam, und man ist mehr als zuvor auf die Hilfe von DNS angewiesen. Im Kapitel 23, »DNS-Server«, beschreiben wir die Konfiguration eines IPv6-DNS-Servers.

IPv6-Adressen werden im Gegensatz zu IPv4-Adressen hexadezimal dargestellt. Jede IPv6-Adresse besteht aus acht Blöcken zu jeweils 16 Bit. Einzelne Blöcke werden durch einen Doppelpunkt getrennt. Eine gültige IP-Adresse sieht also folgendermaßen aus:

```
2001:0db8:0000:0000:0342:abff:fe22:1d63
```

Listing 22.56 Beispiel für eine gültige IPv6-Adresse

Führende Nullen innerhalb eines Blocks dürfen ausgelassen werden:

```
2001:db8:0:0:342:abff:fe22:1d63
```

Listing 22.57 Das Beispiel ohne führende Nullen

Wenn ein oder mehrere Blöcke nur aus Nullen bestehen, so kann man sie durch einen Doppelpunkt ersetzen. Das darf jedoch nur an einer Stelle passieren, weil die Adresse sonst nicht eindeutig ist:

2001:db8::342:abff:fe22:1d63

Listing 22.58 Gekürzte Schreibweise der IPv6-Adresse

Die Adresse für *localhost* lautet so:

0000:0000:0000:0000:0000:0000:0000:0001

Listing 22.59 Die IPv6-Adresse des »localhost«

In der Kurzschreibweise wird daraus:

::1

Listing 22.60 Die Kurzschreibweise der »localhost«-Adresse

22.5.3 Die Netzmasken

Wie IPv4-Adressen bestehen auch IPv6-Adressen aus einem Netzanteil und einem Hostanteil. In der IPv6-Welt spricht man von dem *Network Prefix* und dem *Interface Identifier*. Die Darstellung erfolgt im *Classless Inter-Domain Routing*-(CIDR-)Format. Der Interface Identifier ist mindestens 64 Bit lang.

Kleinere Netze sollten nicht vergeben werden. Die restlichen 64 Bit bilden das *Network Prefix*. Dieses kann aber nach altbekannter Manier in weitere Subnetze aufgeteilt werden (siehe Tabelle 22.3).

Network Prefix	Subnet Identifier	Interface Identifier
48 Bit	16 Bit	64 Bit
2001:0db8:0000	0000	0342:abff:fe22:1d63

Tabelle 22.3 Aufbau der Netzmaske

22.5.4 Die verschiedenen IPv6-Adressarten

Für IPv6 gibt es verschiedene Arten von Adressierungen:

► Unicast

Unicast-Adressen adressieren genau einen Host, wie Sie das schon von IPv4 her kennen. Es gibt hier aber noch eine weitere Unterscheidung:

– global

Die *global*-Adresse dient zur eindeutigen Identifizierung eines Hosts im Netzwerk und wird auch über Router hinweg erreichbar sein. Eine *global*-Unicast-Adresse beginnt immer mit dem Binär-Präfix 001. Als Beispiel sei hier die 2000::/3 genannt.

- **link-local**

Die *link-local*-Adresse darf nur im lokalen Netz verwendet werden und darf nicht geroutet werden. Diese Adresse dient zur Autokonfiguration einer *global*-Adresse in Zusammenarbeit mit dem *Neighbor Discovery*. Mit dieser Adresse können Sie ein temporäres Netzwerk einrichten, wenn Sie zum Beispiel nur kurz Rechner miteinander verbinden müssen, um Daten auszutauschen. Die *link-local*-Adressen beginnen immer mit dem Präfix FE80::8. Ein Beispiel dazu sehen Sie in Listing 22.61:

```
eth0 Link encap:Ethernet Hardware Adresse 00:1d:92:7d:26:a0
      inet Adresse:192.168.123.2 Bcast:192.168.123.255\
        Maske:255.255.255.0
      inet6-Adresse: fe80::21d:92ff:fe7d:26a0/64\
        Gültigkeitsbereich:Verbindung
```

Listing 22.61 Beispiel für eine »link-local«-Adresse

- **unique local**

Bei IPv4 gibt es in den verschiedenen Netzklassen die privaten Adressbereiche, die Sie für Ihre Netzwerke verwenden und die im Internet nicht geroutet werden. Eine ähnliche Adresse gibt es bei IPv6 auch. Das ist die *unique local*-Adresse. Hier werden wieder zwei verschiedene Adressbereiche unterschieden:

- **zentral verwaltet**

Diese Adressen werden vom Provider vergeben. Das Präfix für diese Adressen lautet FC00::/7.

- **lokal verwaltet**

Diesen Bereich können Sie in Ihrem eigenen Netzwerk selbst einteilen. Das Präfix für diese Adressen lautet FD80::/8.

- ▶ **Anycast**

Anycast-Adressen können Sie immer dann verwenden, wenn bei Ihnen im Netzwerk mehrere Router oder Server denselben Dienst bereitstellen sollen. Dabei wird auf jedem Server oder Router ein Interface mit derselben IP-Adresse belegt. Wenn dann ein Client eine Anfrage stellt, kann diese von einem beliebigen Rechner angenommen werden. Sie müssen dabei nur beachten, dass der Absender keine Kontrolle darüber hat, an welches der Systeme die Anfrage gestellt wird. Die Entscheidung trifft das Routing-Protokoll.

- ▶ **Multicast**

Hinter einer *Multicast*-Adresse können sich ganze Gruppen von IPv6-Adressen verbergen. Genau wie bei IPv4 wird auch bei IPv6 ein Datenpaket, das an eine *Multicast*-Adresse gesendet wird, von allen Gruppenmitgliedern empfangen und verarbeitet. Eine *Multicast*-Adresse beginnt immer mit dem Hexadezimalwert 0xFF. Es gibt eine Liste aller schon fest zugewiesenen *Multicast*-Adressen. Diese Liste finden Sie unter www.iana.org/assignments/ipv6-multicast-addresses.

Hier folgen drei Regeln zur Einrichtung eines IPv6-Hosts:

1. Eine Netzwerkkarte hat immer eine *link-local*-Adresse.
2. Einer Netzwerkkarte können Sie immer mehrere globale Adressen zuweisen.
3. Eine Netzwerkkarte kann zu mehreren *Multicast*-Gruppen gehören, mindestens aber zu einer. Im Fall eines einfachen Hosts ist es die *Multicast*-Gruppe FF01::1, im Fall, dass es sich um einen Router handelt, die *Multicast*-Gruppe FF02::2.

Keine Broadcast-Adressen mehr bei IPv6!

Broadcast-Adressen, wie Sie sie von IPv4 her kennen, gibt es bei IPv6 nicht mehr. Diese wurden von den Multicast-Adressen abgelöst.



In Tabelle 22.4 sehen Sie eine kurze Zusammenfassung der reservierten Präfixe.

Präfix	Verwendung
::1	<i>localhost</i>
FE80::/10	<i>link-local</i>
FF00::8	<i>multicast</i>
FF01::1	<i>multicast</i> an alle Hosts
FF02::2	<i>multicast</i> an alle Router
FC00::/8	<i>unique local address</i> (zentral vergeben)
Fd00::/8	<i>unique local address</i> (kann privat verwendet werden)
2000::/3	<i>global unicast address</i>
2001:db8::32	Präfix für Dokumentationen

Tabelle 22.4 Liste der IPv6-Präfixe

22.5.5 Es geht auch ohne ARP

Unter IPv4 ist das Protokoll ARP eine der größten Sicherheitslücken. Unter IPv6 findet das Protokoll ARP keine Verwendung mehr. An dessen Stelle tritt *Neighbor Discovery* (ND). Neighbor Discovery ist ein Bestandteil des *ICMP*-Protokolls unter IPv6 und hat dort mehr Aufgaben, als nur die MAC-Adresse zur IP-Adresse aufzulösen, so wie es ARP unter IPv4 macht. Ein IPv6-Host kann Neighbor Discovery für folgende Aufgaben verwenden:

► **Für die automatische Konfiguration der IPv6-Adressen**

Ein IPv6-Host kann sich selbst eine *link-local*-Adresse geben, um sich automatisch von einem Router eine verfügbare Adresse aus dem lokalen Adressraum zu holen. Über diesen Weg kommt der Host zwar nicht an DNS-Server oder Domainnamen, aber zumindest ist er am Netz.

► **Zur Ermittlung des Standardgateways und des Netzwerkpräfixes**

Durch die automatische Ermittlung des Standardgateways kann ein Host sofort in einem Netzwerk erreicht werden. Auch kann der Host verschiedene Präfixe erhalten und damit auch mehrere IPv6-Adressen beziehen.

► **Doppelte IP-Adressen können ermittelt werden**

Über den ICMP-Typ *Duplicate Address Detection* (DAD) wird ermittelt, ob die IP-Adresse, die automatisch vom Router kommt, schon vergeben ist.

► **Zur Ermittlung der MAC-Adressen aller Hosts im selben Netz**

Besitzt ein Host mehrere Netzwerkkarten mit unterschiedlichen Präfixen, werden für jede Netzwerkkarte die MAC-Adressen in diesem Präfix ermittelt.

► **Zur Suche der Router an derselben Netzwerkschnittstelle**

Über Neighbor Discovery wird die automatische Konfiguration durchgeführt. Dieses geschieht immer über einen Router. Bevor die automatische Konfiguration durchgeführt werden kann, wird über Neighbor Discovery der entsprechende Router gesucht.

► **Aktualisierung der Router an der Schnittstelle**

Sollte sich ein Router im Netz ändern, kann diese Information automatisch in die Konfiguration übernommen werden.

► **Einsatz zur Ermittlung von erreichbaren und nicht erreichbaren Hosts**

Jeder Host pflegt eine Liste seiner Nachbarn im *Neighbor Cache*. Diese Liste ist in etwa vergleichbar mit der *ARP-Table* unter IPv4. In dieser Liste wird gespeichert, ob es sich um einen normalen Host oder um einen Router handelt. Hier wird auch gespeichert, ob ein Host erreichbar ist oder nicht. Es wird ebenfalls gespeichert, wann der nächste Test auf Erreichbarkeit stattfinden muss.

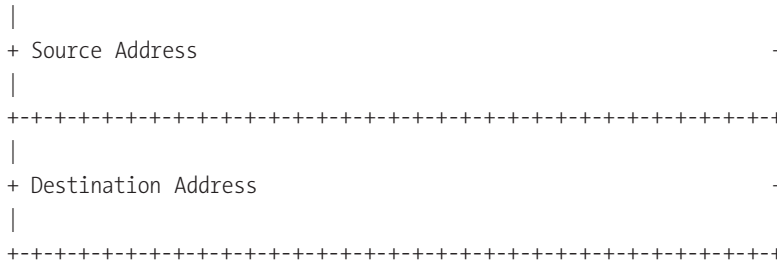
22.5.6 Feste Header-Länge

Die Kopfdatenlänge eines IPv6-Pakets beträgt immer genau 40 Byte. IPv4 hingegen arbeitet mit einer variablen Header-Länge. Eine feste Länge hat für Router den Vorteil, dass die Verarbeitung weniger rechenintensiv ist. Listing 22.62 zeigt den Aufbau des IPv6-Headers:

```

+++++
|Version| Traffic Class |      Flow Label      |
+++++
| Payload Length      | Next Header  | Hop Limit    |
+++++

```



Listing 22.62 Aufbau des IPv6-Headers

Die einzelnen Felder haben dabei folgende Bedeutungen:

- ▶ **Version**
gibt die IP-Version in 4 Bit an, hier also die 6.
- ▶ **Traffic Class**
ein 8 Bit langes Feld, das dem *Type of Service*-Feld im IPv4-Header entspricht. Mithilfe dieses Feldes ist es einfacher, Realtime-Daten zu priorisieren.
- ▶ **Flow Label**
ein 20 Bit langes Feld, mit dem die Zusammengehörigkeit verschiedener Pakete gekennzeichnet werden kann. Alle Pakete mit demselben Wert werden vom Router gleich behandelt. Das erleichtert und beschleunigt die Weiterleitung von priorisierten Daten wie Audio- oder Videodaten. Wichtig ist nur, dass alle Pakete eines *Flows* dieselbe Sender- und Empfängeradresse haben.
- ▶ **Payload Length**
Das Feld hat eine Länge von 16 Bit und gibt die Menge an Daten an, die sich im IPv6-Paket befinden.
- ▶ **Next Header**
Dieses Feld hat eine Länge von 16 Bit und entspricht dem *Protocol Type*-Feld im IPv4-Header. Bei IPv6 kann dieses Feld auf einen nächsten Protokoll-Header zeigen oder auf einen Extension-Header. In Extension-Headern werden zusätzliche Funktionen wie Fragmentierung, Routing-Informationen oder Authentifizierungen gesendet.
- ▶ **Hop Limit**
Dies ist ein 8 Bit langes Feld, das dem *TTL* im IPv4-Header entspricht. Genau wie bei IPv4 reduziert jeder Router den Wert um 1; wenn der Wert bei 0 angekommen ist, wird der Router das Paket verwerfen und dem Absender eine Nachricht schicken.
- ▶ **Source Address**
die 128 Bit lange Senderadresse
- ▶ **Destination Address**
die 128 Bit lange Empfängeradresse

Multihoming

Mit IPv6 ist es ohne größere Probleme möglich, mit einem Netz an mehrere Provider angebunden zu sein. Das vereinfacht das Umziehen von Netzen und kann genutzt werden, um eine erhöhte Ausfallsicherheit zu erreichen. Der *Interface Identifier* muss im sogenannten *Modified EUI-64-Format* vorliegen. Im Wesentlichen bedeutet das, dass bei Ethernet aus der 48-Bit-MAC-Adresse eine 64-Bit-Adresse generiert wird. Der Vorteil dieses Verfahrens ist auch gleichzeitig der Nachteil und einer der größten Kritikpunkte an IPv6. Ein so generierter Interface Identifier ist aufgrund der MAC-Adresse nämlich weltweit einzigartig. Ein Endgerät ist also auch völlig unabhängig von dem *Network Prefix* eindeutig identifizierbar. Aus diesem Grund wurde der Standard später um die *Privacy Extension* für automatische Konfigurationen ergänzt. Die Privacy Extension sorgt dafür, dass der Interface Identifier zufällig generiert und regelmäßig erneuert wird. Das Verhalten muss jedoch zunächst im *proc*-Dateisystem aktiviert werden. Näheres finden Sie in Abschnitt 3.4, »Alles virtuell? >/proc«.

```
root@saturn:~# echo 2 > /proc/sys/net/ipv6/conf/eth0/use_tempaddr
```

Listing 22.63 Aktivierung der »Privacy Extension«

Der Wert 0 steht für die Deaktivierung der Privacy Extension und ist die Standardeinstellung. Die Werte 1 oder 2 aktivieren die Erweiterung. Dabei sorgt 1 dafür, dass bevorzugt öffentliche Adressen verwendet werden, und die 2 besagt, dass eher temporäre Adressen genutzt werden sollen. Die Gültigkeitsdauer in Sekunden können Sie in der Datei */proc/sys/net/ipv6/conf/eth0/temp_valid_lft* eintragen.

22.5.7 IPv6 in der Praxis

Es wäre sicherlich falsch, zu behaupten, dass IPv6 genau wie IPv4 funktioniert, aber zumindest aus Applikationssicht ist es nicht ganz unwahr. Sind die Dienste nämlich erst einmal konfiguriert, so ist kein großer Unterschied festzustellen. Viele Applikationen binden sich heute schon standardmäßig auf alle vorhandenen IPv6-Adressen, sodass nicht immer Konfigurationsanpassungen vorzunehmen sind (zum Beispiel bei *ssh* oder Webservern).

```
root@ubuntu:~# ssh 2a01:198:689::1
root@2a01:198:689::1's password:
[...]
root@ubuntu:~# telnet 2a01:198:689::1 80
Trying 2a01:198:689::1...
Connected to 2a01:198:689::1.
Escape character is '^]'.
GET /
<html><body><h1>It works!</h1>
[...]
```

Listing 22.64 »ssh« und Web mit »telnet« über IPv6

Etwas anders sieht es aus, wenn Sie nur *link-local*-Adressen aus dem Netz `fe80::/10` benutzen. Diese werden in der Regel durch die automatische Konfiguration auf das Interface gebunden. Die Art der Adresse können Sie nicht nur an dem Präfix erkennen, sondern auch, indem Sie sich den *scope* bei der Ausgabe von `ip` anschauen:

```
root@saturn:~# ip -f inet6 address show dev eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
    inet6 2a01:198:689::1/64 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::226:18ff:fe81:1d63/64 scope link
        valid_lft forever preferred_lft forever
```

Listing 22.65 »link local«- und globale IPv6-Adresse auf »eth0«

Link-local-Adressen können mehrfach auf einem Gerät vergeben werden, daher müssen Sie angeben, über welches Interface die Anfrage gehen soll. Bei *ssh* geschieht das beispielsweise, indem Sie hinter der IP-Adresse `%INTERFACENAME` anhängen:

```
root@ubuntu:~# ssh fe80::226:18ff:fe81:1d63%eth0
root@fe80::226:18ff:fe81:1d63%eth0's password:
[...]
```

Listing 22.66 »ssh« über eine »link local«-Adresse

Zum Testen der Konnektivität können Sie übrigens *ping6* nutzen:

```
root@saturn:~# ping6 -c 1 ipv6.test-ipv6.com

PING ipv6.test-ipv6.com(2001:470:1:18::119) 56 data bytes
64 bytes from 2001:470:1:18::119: icmp_seq=1 ttl=58 time=34.9 ms
--- ipv6.test-ipv6.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 34.965/34.965/34.965/0.000 ms
```

Listing 22.67 »ping« auf »google« per IPv6

Selbstverständlich sind nicht alle Dienste sofort für IPv6 nutzbar. Sie müssen in der jeweiligen Dokumentation nachlesen, was an welcher Stelle konfiguriert werden muss. In den meisten Fällen reicht es aber aus, dem Dienst einfach mitzuteilen, dass er vorhandene IPv6-Adressen nutzen soll.

Bei einer Umstellung auf IPv6 sollten Sie unbedingt zunächst eine Firewall mit *ip6tables* konfigurieren und dann die entsprechenden Einträge im DNS anpassen. Mehr zu diesen Themen finden Sie in Abschnitt 22.6, »Firewalls mit ›netfilter‹ und ›iptables‹«, und Kapitel 23, »DNS-Server«.



22.6 Firewalls mit netfilter und iptables

In den meisten Fällen wird bei Linux, wenn es um Firewalls geht, von *iptables* gesprochen – aber eigentlich ist *netfilter* gemeint, da dies hinter dem Frontend *iptables* steht. *netfilter* ist die Schnittstelle im Kernel, mit der sich Pakete abfangen und manipulieren lassen. *iptables* hingegen ist nur ein Tool, mit dem *netfilter* angesprochen werden kann. Für andere Schichten und Protokolle gibt es noch *ebtables*, *ip6tables* und *arptables*. In diesem Abschnitt zeigen wir Ihnen die Möglichkeiten von *iptables*. Sowohl der Ablauf als auch die Syntax und die Kommandos lassen sich fast 1:1 auf *ip6tables* (das entsprechende Pendant für IPv6-Umgebungen) übertragen.

22.6.1 Der Weg ist das Ziel – wie Pakete durch den Kernel laufen

Mit *iptables* werden Filterregeln gesetzt, die darüber entscheiden, was mit einem Paket passiert. Diese Regeln werden in unterschiedliche Ketten, die sogenannten *Chains*, geschrieben. Je nach Ursprung und Ziel eines Pakets werden unterschiedliche Chains durchlaufen. Abbildung 22.3 zeigt den Verlauf eines Pakets im Kernel.

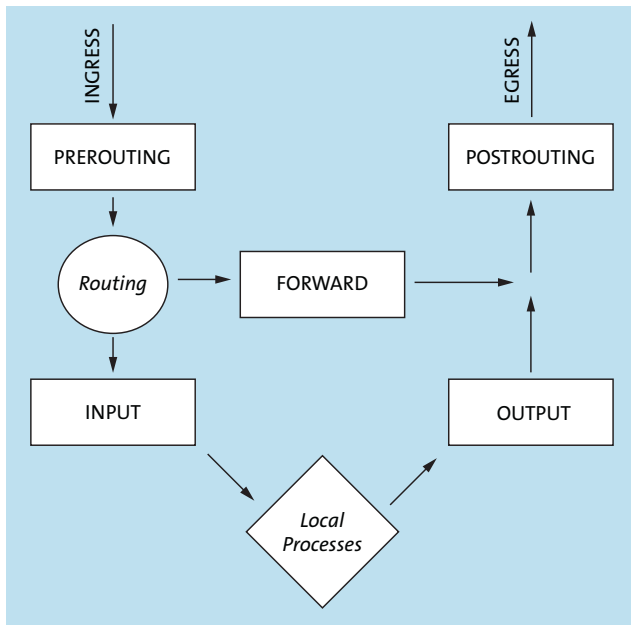


Abbildung 22.3 Paketverlauf im Kernel

Alle Pakete, die das System erreichen, durchlaufen zunächst die *PREROUTING*-Chain. Wie der Name bereits sagt, wurde zu diesem Zeitpunkt noch keine Routing-Entscheidung getroffen. Daher ist auch noch nicht bekannt, ob das Paket an das System adressiert ist oder den Rech-

ner nur als Router durchläuft. Aus diesem Grund werden in der *PREROUTING*-Chain auch keine klassischen Filterregeln gesetzt, sondern beispielsweise Regeln für *Network Address Translation* (NAT).

Nach der *PREROUTING*-Chain wird die Routing-Entscheidung getroffen. Falls das Paket an das System adressiert ist, wird als Nächstes die *INPUT*-Chain angesprochen. Daher werden in der *INPUT*-Chain auch Regeln definiert, die sich auf eingehende Pakete beziehen. Nach erfolgreichem Durchlauf kann das Paket dann an den lokalen Prozess zugestellt werden. Dies kann zum Beispiel ein Webserver sein, der lokal auf der Maschine läuft. Die Antworten des Prozesses werden in die *OUTPUT*-Chain eingeliefert. Der ausgehende Datenverkehr kann daher ebenfalls gefiltert werden. Zu guter Letzt wird das Paket an die *POSTROUTING*-Chain übergeben. Hier können Pakete analog zum *PREROUTING* manipuliert werden.

Pakete, die von dem System geroutet werden, durchlaufen weder die *INPUT*- noch die *OUTPUT*-Chain, sondern die *FORWARD*-Chain. Dort werden also alle Regeln gesetzt, um ein dahinter liegendes Netzwerk zu schützen.

Jede Chain besteht aus mehreren *tables* (zu Deutsch *Tabellen*). Aber nicht jede Tabelle ist in allen Chains vorhanden. Wenn in *iptables* nicht explizit eine Tabelle angegeben wird, so wird automatisch die *filter*-Tabelle verwendet. Folgende Tabellen können von *iptables* genutzt werden:

► **filter**

Wie der Name vorgibt, werden in dieser Tabelle klassische Paketfilterregeln erzeugt. Die *filter table* ist in der *INPUT*-, *OUTPUT*- und *FORWARD*-Chain zu finden.

► **nat**

Für *Network Address Translation* gibt es die *nat*-Tabelle. Diese existiert nur in *PREROUTING*, *OUTPUT* und *POSTROUTING*.

► **mangle**

In der *mangle*-Tabelle werden Pakete manipuliert, was wir bereits in Abschnitt 22.3.2, »Da geht noch mehr: ›Advanced Routing‹«, durchgeführt haben. Diese Tabelle ist in jeder Chain zu finden.

► **raw**

Die letzte Tabelle ist die Tabelle *raw* (engl. für *roh* oder *unbearbeitet*). Diese Tabelle existiert ausschließlich im *PREROUTING* und *OUTPUT*. Sie kann genutzt werden, um Pakete gezielt am *Connection Tracking* vorbeizuschleusen – also um die Pakete unbearbeitet von bestehenden Verbindungen aus zu betrachten.

22.6.2 Einführung in iptables

Mit *iptables* können Sie nicht nur die Firewallregeln definieren, sondern auch die Chains verwalten. Diese Möglichkeiten wollen wir Ihnen nun der Reihe nach vorstellen.

Ausgabe der Regeln: -L <CHAIN> [-t <TABLE>]

Mit der Option `-L` können Sie sich alle Regeln ausgeben lassen. Ohne weiteren Parameter werden alle Regeln der Tabelle *filter* von allen Chains ausgegeben. Geben Sie eine Chain an, werden nur deren Regeln der Tabelle *filter* ausgegeben. In Kombination mit der Option `-t <TABLE>` können auch andere Tabellen ausgegeben werden.

**Zusätzliche Angaben: »-n« und »-v«**

Sie sollten zusätzlich zu `-L` auch noch `-n` und `-v` angeben. Mit `-n` unterbinden Sie die Namensauflösung der IP-Adressen und sparen somit bei großen Listen viel Zeit. `-v` gibt zusätzliche wichtige Statistiken mit an, und darüber hinaus wird die *Default Policy* der Chain mit angezeigt.

In Listing 22.68 sehen Sie eine Ausgabe der Auflistung aller Chains:

```
Chain INPUT (policy ACCEPT 19 packets, 1406 bytes)
  pkts bytes target    prot opt in     out     source            destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target    prot opt in     out     source            destination

Chain OUTPUT (policy ACCEPT 8 packets, 1120 bytes)
  pkts bytes target    prot opt in     out     source            destination
```

Listing 22.68 Auflistung aller Chains

Regeln löschen: -F <CHAIN> [-t <TABLE>]

Mit der Option `-F` können Sie alle Regeln aus einer Chain und deren Tabellen löschen. Ohne weitere Parameter werden lediglich die Regeln der *filter*-Tabelle entfernt.

**Beim Löschen nicht »mangle«, »raw« und »nat« vergessen!**

Die Tabellen *mangle*, *raw* und *nat* müssen Sie explizit angeben, um deren Inhalt zu löschen. Wenn Sie mit diesen Tabellen arbeiten, dürfen Sie das nicht vergessen! Da diese Tables mit `-L` im Normalfall unterschlagen werden, vergisst man sie häufig. Unter Umständen bleiben dadurch noch Reste einer alten Konfiguration aktiv.

Angabe der Default-Chain: -P <CHAIN> <TARGET>

Mit der Option `-P` wird für die angegebene Chain die *Default-Policy* gesetzt. Diese wird auch oft als *Clean Up Rule* bezeichnet, da sie immer zutrifft, wenn vorher keine andere Regel zugegriffen hat – sie ist sozusagen das Aufräumkommando. Über `<TARGET>` wird das Standardver-

halten angegeben. (Mehr zum Thema TARGET finden Sie in Abschnitt 22.6.4, »Die klassischen Targets«.)

Nach dem Systemstart steht die *Default-Policy* auf ACCEPT – daher werden alle Pakete, für die keine expliziten Regeln existieren, einfach zugelassen. Die *Default-Policy* kann nur auf die vordefinierten Chains gesetzt werden, nicht auf selbst erstellte.

Beim Löschen nicht enthalten!

Beim Löschen aller Regeln mittels `-F` wird die *Default-Policy* auf ACCEPT zurückgesetzt! Beachten Sie dies, bevor Sie Regeln löschen, damit Sie keine ungewollte Kommunikation zulassen.



Anlegen und Löschen von Chains: `-N <CHAIN>` und `-X <CHAIN>`

Mit der Option `-N` werden eigene Chains angelegt. Die maximale Länge für einen Namen beträgt im Übrigen 30 Zeichen. Verwenden Sie nur Klein- und Großbuchstaben, Zahlen, den Binde- und Unterstrich. Vermeiden Sie Sonderzeichen (auch deutsche Umlaute), da dies zu Fehlern führen kann.

Äquivalent zum Anlegen können Sie mit der Option `-X` eine Chain angeben, die gelöscht werden soll.

Anlegen und Bearbeiten von Regeln: `-A <CHAIN>`, `-I <CHAIN> [<POS>]` und `-D <CHAIN> <POS/RULE>`

Mit der Option `-A` können Sie eine Regel in der angegebenen Chain anlegen. Dabei wird diese Regel als letzte Regel zur Chain hinzugefügt.

Die Option `-I` erwartet zusätzlich zur Chain die Position, an der die neue Regel hinzugefügt werden soll. Die Position geben Sie dabei als Zeilennummer an. Falls Sie nicht stumpf die Zeilen abzählen wollen, können Sie auch den Befehl `iptables -L --line-numbers` absetzen. Dieser gibt Ihnen die bestehenden Regeln mit vorangestellter Zeilennummer aus. Geben Sie keine Position an, wird die Regel an erster Stelle der Chain hinzugefügt.

Zum Entfernen einer Regel können Sie die Option `-D` verwenden. Dabei müssen Sie entweder die entsprechende Regel vollständig angeben, oder Sie geben erneut, wie bereits beim Einfügen mittels `-I`, die Zeilennummer der entsprechenden Regel an.

22.6.3 Regeln definieren

Regeln bestehen aus einem oder mehreren Filtern, die zum einen das Paket beschreiben und zum anderen das Ziel (*target*) beinhalten, das angibt, was mit dem Paket passieren soll. Folgende Filter können Sie verwenden:

- ▶ `-i / -o <INTERFACE>`
bezeichnet das Input- bzw. Output-Interface, über das das Paket geführt wird. Die Filter können natürlich nicht in jeder Chain eingesetzt werden. Die Option `-i` kann in `PREROUTING`, `INPUT` und `FORWARD` genutzt werden, die Option `-o` hingegen in `POSTROUTING`, `OUTPUT` und `FORWARD`.
- ▶ `-s / -d <IP-ADDRESS>`
Mit den Optionen `-s` und `-d` können die Quelle (*source*) und das Ziel (*destination*) angegeben werden. Dabei versteht *iptables* sowohl einzelne IP-Adressen als auch Netze in der CIDR-Notation.
- ▶ `-p <PROTOCOL>`
Die Option `-p` definiert das Protokoll für die Regel. Zum Beispiel können Sie *tcp*, *udp* oder *icmp* angeben.
- ▶ `--dport <PORT[:PORT]> / --sport <PORT[:PORT]>`
Diese Filter sind eigentlich modulspezifische Erweiterungen für *tcp* und *udp*. Da sie aber zu den gängigsten Filtern gehören, führen wir sie hier direkt mit auf. Mit diesen Optionen können Sie auf Ziel- und Quellports filtern. Dabei versteht *iptables* sowohl einzelne Ports als auch Portbereiche (diese werden durch einen Doppelpunkt getrennt angegeben). Die Angabe von Ports muss nicht in Ziffern erfolgen, Sie können auch die Namen aus der Datei */etc/services* verwenden.
- ▶ `-m <MODUL>`
Mit der Option `-m` können Sie weitere *iptables*-Module nachladen, die zusätzliche Filtermöglichkeiten anbieten. Einige Module werden implizit bei der Angabe eines Protokolls geladen (zum Beispiel wird das *icmp*-Modul automatisch bei Verwendung von *icmp* geladen). Die Module werden in der Manpage beschrieben. Alternativ können Sie auch `iptables -m <MODUL> --help` aufrufen. Nach der Kurzbeschreibung des *iptables*-Befehls werden anschließend die modulspezifischen Erweiterungen angezeigt.



Mehrfachverwendung

Wenn Sie mehr als ein Modul nachladen möchten, können Sie die Option `-m` mehrfach angeben.

- ▶ `-j <TARGET>`
Mit der Option `-j` wird das Ziel der Regel angegeben. Es wird also angegeben, was mit dem passenden Paket passieren soll.



Pakete durchlaufen nacheinander die beschriebenen Chains. Die meisten Targets sind terminierend. Das heißt, sie entscheiden sofort, was mit dem Paket passieren soll. Daher greift immer die erste zutreffende Regel – somit ist die Reihenfolge der Regel von essenzieller Wichtigkeit.

Zusätzlich zur Funktionalität spielt die Reihenfolge auch eine große Rolle bei der Performance. *netfilter* ist zwar sehr performant, aber bei Regelwerken mit vielen und komplexen Regeln kann sich eine nicht optimal gewählte Reihenfolge durchaus bemerkbar machen. Regeln, die viele Pakete verarbeiten, sollten daher immer möglichst weit oben im Regelwerk stehen. Falls Sie viele Webserver betreiben, sollte die Regel für HTTP entsprechend weit oben in Ihrem Regelwerk einsortiert werden und nicht erst an Position 1.000.

22.6.4 Die klassischen Targets

Wie wir bereits erörtert haben, entscheiden Targets darüber, was mit einem Paket geschieht, das für eine Regel zutrifft. Nachstehend haben wir die möglichen Targets aufgelistet:

- ▶ ACCEPT
Das Paket wird durchgelassen.
- ▶ DROP
Das Paket wird kommentarlos verworfen. Der anfragende Rechner läuft also mit der Anfrage in einen Timeout.
- ▶ REJECT
Das Paket wird verworfen, allerdings bekommt der Absender eine Fehlermeldung über *icmp* zurück.

22.6.5 Ein erster Testlauf

In diesem Abschnitt wollen wir Ihnen zeigen, wie Sie das bisher erworbene Wissen in der Praxis anwenden können. Listing 22.69 zeigt, wie Sie eine Webseite mit iptables sperren:

```
root@saturn:~# iptables -A OUTPUT -p tcp --dport http \
                -d www.rheinwerk-verlag.de -j REJECT
root@saturn:~# telnet www.rheinwerk-verlag.de 80
Trying 46.235.24.168...
telnet: Unable to connect to remote host: Connection refused

root@saturn:~# iptables -vnL OUTPUT
Chain OUTPUT (policy ACCEPT 2356 packets, 198K bytes)
pkts bytes target prot opt in  out  source  destination
2    120 REJECT  tcp  --  *   *    0.0.0.0/0  46.235.24.168  tcp dpt:80\
                reject-with icmp-port-unreachable
```

Listing 22.69 Sperren einer Webseite mit »iptables«

Die Regel aus Listing 22.69 blockiert (REJECT) den ausgehenden (-A OUTPUT) HTTP-Verkehr (--dport http) auf den Webserver des Rheinwerk Verlags (-d www.rheinwerk-verlag.de). Da wir als Target REJECT gewählt haben, wird die Verbindung abgewiesen – was durch die Rück-

meldung `Connection refused` von `telnet` dargestellt wird. Hätten wir `DROP` gewählt, würde `telnet` bis zum `Timeout` gewartet haben, bevor es aufgegeben hätte. Abschließend haben wir die Informationen zur `OUTPUT-Chain` abgefragt. Der Ausgabe können Sie entnehmen, dass die Regel bisher für zwei Pakete mit insgesamt 120 Byte zugetroffen hat.



Bei der Regel aus Listing 22.69 haben wir einen DNS-Namen anstelle einer IP-Adresse verwendet. Dies ist durchaus legitim und kann sehr praktisch sein, vor allem, wenn sich hinter einem Namen mehrere IP-Adressen verbergen. Allerdings müssen Sie dabei beachten, dass `iptables` die Namensauflösung durchführt, bevor es die Regel in den Kernel schreibt. Ändert sich anschließend die Zuordnung von Namen zur IP-Adresse, müssen Sie die Regel erneut setzen – die Namensauflösung findet also nicht bei jedem Zugriff (oder nach Ablauf der Gültigkeitsdauer des DNS-Eintrags) statt!

22.6.6 Rein wie raus: Stateful Packet Inspection

Die *Stateful Packet Inspection* (SPI) kontrolliert nicht nur Quelle, Ziel und Protokoll, sondern prüft auch den Zustand einer Verbindung. Dabei wird geprüft, ob ein Paket zum Aufbau einer Verbindung verwendet wird oder ob es zu einer bereits bestehenden Verbindung gehört. Der Kernel führt dafür mit dem *Connection Tracking* eine Liste der bestehenden Verbindungen. Mit den bisher vorgestellten Optionen kann diese Funktion nicht eingerichtet werden. Um zum Beispiel von internen Systemen aus den Zugriff auf einen Webserver zu erlauben, müssten Sie die Regel aus Listing 22.70 einrichten:

```
root@saturn:~# iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
root@saturn:~# iptables -A INPUT -p tcp --sport 80 -j ACCEPT
```

Listing 22.70 Zugriff auf Port 80 freischalten

Diese Regel funktioniert zwar, ist aber zum einen unschön, und zum anderen provoziert sie auch ein Verhalten, das meistens nicht erwünscht ist. Unschön ist die Regel vor allem, da wir für eine Verbindung zwei Regeln einrichten mussten. Darüber hinaus besagen die Regeln lediglich, dass Daten aus dem internen Netzwerk an Systeme über Port 80 gesendet werden dürfen und dass Antworten von Port 80 zugelassen werden – diese Regeln haben keine Verbindung, und daher ist nicht sichergestellt, dass nur Antworten auf vorherige Anfragen zugelassen werden. Somit könnte das interne System A eine Anfrage an den Server C stellen, und völlig unabhängig davon könnte Server B Daten in das interne Netzwerk senden. Die wahre Krux besteht aber darin, dass nicht alle Protokolle immer mit dem gleichen Port antworten – in der Regel spricht der Client den Server auf einem definierten Port an, und während des Kommunikationsaufbaus handeln beide den gültigen Rückkanal aus, meist einen beliebigen *High Port* (Port > 1024).

Dank SPI müssen Sie nicht unnötig viele Regeln erzeugen, um Datenverkehr gezielt freigeben zu können. Damit `iptables` die SPI verwendet, müssen Sie das Modul `state` laden. Dieses

Modul bietet Ihnen nun die zusätzliche Option `--state` an. Diese Option kann nachstehende Parameter verarbeiten:

- ▶ NEW
Das Paket startet eine neue Verbindung. Bei TCP-Datenverkehr ist das also das initiale Paket (mit gesetztem *syn*-Flag).
- ▶ ESTABLISHED
Das Paket gehört zu einer bestehenden Verbindung – es wurden also bereits Daten in beide Richtungen ausgetauscht.
- ▶ RELATED
Das Paket baut eine neue Verbindung auf, allerdings wurde diese Verbindung infolge einer bereits existierenden Verbindung erzeugt. Diese Sonderform wird beispielsweise bei FTP benötigt, da FTP zunächst eine Kontrollverbindung über *tcp/21* aufbaut und anschließend zum Datenaustausch *tcp/20* verwendet. Um diese Funktion nutzen zu können, müssen Sie zusätzlich das entsprechende Kernelmodul mittels `modprobe nf_conntrack_ftp` laden. Neben FTP werden weitere Protokolle mit ähnlichem Verhalten von diesem Kernelmodul unterstützt.
- ▶ INVALID
Es konnte nicht bestimmt werden, zu welcher Kategorie das Paket gehört – dieser Zustand kommt zum Beispiel vor, wenn ein Paket an eine nicht vorhandene Verbindung gesendet wird.

Bei TCP-Verbindungen kann, da das Protokoll zustandsorientiert ist, leicht festgestellt werden, ob ein Paket zu einer bestehenden Verbindung gehört. Aber auch beim zustandslosen UDP ist *netfilter* in der Lage, den Zustand festzustellen. Dafür werden einfach die Rahmenbedingungen (wie zum Beispiel Quelle, Ziel, Portnummern und auch die Zeit) zur Zuordnung hinzugezogen. Das *Connection Tracking* kann daher unabhängig vom eingesetzten Protokoll verwendet werden und stellt ein großes Hilfsmittel bei der Erstellung effektiver und übersichtlicher Regelwerke dar.

Das Regelwerk aus Listing 22.70 kann mit SPI so abgebildet werden, wie in Listing 22.71 dargestellt:

```
root@saturm:~# iptables -A OUTPUT -p tcp -m state --state NEW,ESTABLISHED \  
    > --dport 80 -j ACCEPT  
root@saturm:~# iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
```

Listing 22.71 Regel für Port 80 mit »SPI«

Wie Sie Listing 22.71 entnehmen können, wird bei den Regeln nun mit der Option `-m` das Modul `state` geladen. Daher kann über die Option `--state` auch der für die Regel erlaubte Zustand angegeben werden. Die erste Regel lässt also ausgehende (`-A OUTPUT`) Pakete auf

`tcp/80` für neue (NEW) und bereits etablierte (ESTABLISHED) Verbindungen zu. Die zweite Verbindung erlaubt nur Antwortpakete an bereits etablierte Verbindungen.

In der Praxis wird fast ausschließlich mit SPI gearbeitet. Nicht nur, weil dadurch die Sicherheit erhöht wird, sondern auch, weil die Übersichtlichkeit der Regelwerke gesteigert wird. Die Steigerung der Übersichtlichkeit wird vor allem daran deutlich, dass Sie nur für die drei Filter-Chains je eine Regel für die Zustände RELATED und ESTABLISHED einrichten müssen, um alle möglichen Antworten an bestehende Verbindungen zuzulassen.

Daher reicht es, bei so eingerichteten Regeln anschließend nur noch Regeln für den Zustand NEW einzurichten. In Listing 22.72 sehen Sie die Regeln, über die alle Antworten zu bestehenden Verbindungen bei allen Chains automatisch zugelassen werden:

```
root@satur:~# iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
root@satur:~# iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
root@satur:~# iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Listing 22.72 Freischaltung aller Antworten für bestehende Verbindungen

22.6.7 Das erste Firewallskript

iptables-Regelwerke werden in Skripten organisiert, da sie darüber leicht beim Systemstart geladen werden können. Zum Einstieg erstellen wir ein Skript für einen Host. Diesem werden sowohl Web- als auch DNS-Zugriffe in Richtung Internet erlaubt. Zusätzlich sollen aus- und eingehende Pings freigeschaltet werden. Erstellen Sie dafür ein Skript mit dem Inhalt aus Listing 22.73:

```
#!/bin/bash
IPT="/sbin/iptables"

# Setzen der Default-Policy
$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

# Loeschen evtl. vorhandener alter Regeln
$IPT -F

# Loopback freischalten
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

# Bestehende Verbindungen erlauben
$IPT -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
$IPT -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
# Ausgehend HTTP(S) erlauben
$IPT -A OUTPUT -m state --state NEW -m multiport -p tcp --dport http,https -j ACCEPT

# DNS-Abfragen zulassen
$IPT -A OUTPUT -m state --state NEW -p udp --dport domain -j ACCEPT
$IPT -A OUTPUT -m state --state NEW -p tcp --dport domain -j ACCEPT

# Aus- und eingehend ping erlauben
$IPT -A OUTPUT -m state --state NEW -p icmp --icmp-type echo-request -j ACCEPT
$IPT -A INPUT -m state --state NEW -p icmp --icmp-type echo-request -j ACCEPT
```

Listing 22.73 Ein erstes Firewallskript

Wie Sie Listing 22.73 entnehmen können, besteht das Skript aus acht Blöcken, die jeweils mit einer Raute eingeleitet werden. Sehen wir uns diese etwas genauer an.

Im ersten Block wird der Shebang gesetzt und die Variable IPT auf den Programmpfad von iptables gesetzt – passen Sie dieses gegebenenfalls an Ihre Umgebung an. Der zweite Block setzt zunächst die *Default Policy* auf DROP, sodass der nicht von uns freigegebene Datenverkehr verworfen wird – dieser Block gehört quasi zum Standard, da bei jedem Neuladen des Regelwerks die *Default-Policy* von iptables auf ACCEPT gesetzt wird.

Im dritten Block werden etwaige Reste aus den Chains gelöscht, damit wir nicht in einen unbekanntem Zustand hinein neue Regeln definieren. Der vierte Block gibt den Zugriff vom und auf das Loopback-Interface frei. (Dies schauen wir uns im Anschluss noch genauer an.)

Der fünfte Block gehört ebenfalls zum Standardrepertoire eines Regelwerks, da dort die generelle Freisaltung für gültige (also zu bestehenden Verbindungen gehörende) Antwortpakete gesetzt wird. Im fünften, sechsten und siebten Block werden nun endlich die eigentlichen Freigaben eingerichtet. Dort wird das zusätzliche Modul `multiport` verwendet.

Neben den bisher vorgestellten Standardregeln haben wir im Firewallskript zwei Besonderheiten verwendet, die wir Ihnen nun näher erläutern möchten:

1. Loopback-Interface freischalten

Nach dem Setzen der *Default-Policy* im zweiten Block werden sämtliche Verbindungen verboten. Damit werden auch Pakete vom *Loopback-Interface* (`lo`) unterbunden, was sehr häufig vergessen wird. Wenn über `lo` keine Daten mehr ausgetauscht werden können, funktionieren einige Dienste überhaupt oder teilweise nicht. Daher gehört dieser Block zum Standardrepertoire eines Regelwerks.

2. Multiport-Modul

Dieses Modul ermöglicht es Ihnen, eine kommaseparierte Liste von Ports mit den Optionen `--dport` und `--sport` anzugeben. Die Optionen können sonst nur einzelne Ports oder Portbereiche verarbeiten. Das Modul `multiport` erhöht die Übersichtlichkeit enorm.



Es gibt seit jeher die Diskussion, ob Paketfilter standardmäßig mit DROP oder REJECT arbeiten sollten. Für beide Varianten gibt es gute Gründe. Leider existieren auch eine Menge falscher Gründe.

Einer wäre z.B., dass behauptet wird, dass Portscans durch die Verwendung von DROP verlangsamt werden. Das ist aber nur zum Teil korrekt. Ein Portscanner wartet in der Regel nicht den Timeout ab. Das heißt, die Verlangsamung ist marginal bis gar nicht existent. Bei intelligent ausgeführten Portscans wird der Timeout so gewählt, dass er für einen bestimmten Zielhost nicht langsamer ist, egal ob nun das System oder eine Firewall mit einer REJECT-Policy antwortet. Die Einzigen, die in der Regel in Timeouts laufen, sind meist Anwender – was die Usability nicht gerade erhöht.

Ein weiteres gern verwendetes Argument für die Nutzung von DROP ist, dass darüber verschleiert werden kann, welche Dienste ein System anbietet. Leider ist auch dieser Ansatz nicht korrekt, da insbesondere das Verwerfen eines Pakets darauf hindeutet, dass eine Firewall im Spiel ist. Ansonsten würde das angesprochene System eine entsprechende ICMP-Meldung an das anfragende System zurücksenden (was im Übrigen im RFC1122 auch vorgeschrieben wird). Daher ist der einzig wahre Grund, DROP zu verwenden, der, nicht als ICMP-Schleuder missbraucht werden zu können. Falls ein Angreifer mit gefälschten Absenderadressen agiert, würde der eigentliche Besitzer der Adressen von Ihrer Firewall mit ICMP-Meldungen bombardiert werden.



Im Übrigen wurde im Skript aus Listing 22.73 kein REJECT gewählt, weil es nicht als *Default-Policy* gesetzt werden kann.

Falls Sie dennoch ein REJECT vorziehen, müssen Sie als letzte Regel im Skript iptables -A <CHAIN> -j REJECT angeben. Sie müssen als Administrator selbst entscheiden, welcher Mechanismus Ihnen lieber ist oder eher für Ihre Umgebung zutrifft. Wir empfehlen Ihnen, zumindest für das interne Netz REJECT zu verwenden, denn dann laufen wenigstens Ihre Anwender nicht in Timeouts und die Fehlersuche wird erleichtert.

22.6.8 Externe Firewall

Bisher haben wir iptables lediglich in einer relativ flachen Umgebung genutzt. Um zum Beispiel Netzwerke voneinander zu trennen, wie das Internet, eine *Demilitarized Zone* (DMZ) oder das interne Netz (LAN), muss deutlich mehr Aufwand betrieben werden.

Zunächst besteht der größte Unterschied darin, dass wir mit der *FORWARD*-Chain arbeiten müssen. Dort verfügen wir aber nicht über die Richtungsinformationen – also von wo nach wo das Paket fließt, was in den Chains *INPUT* und *OUPUT* vorhanden ist. Da dies aber für die Regeln von entscheidender Bedeutung ist, da wir nun ja mehrere Netze verwalten, müssen wir eine Möglichkeit finden, an diese Information zu gelangen.

Eine Möglichkeit wäre, bei jeder Regel mit `-i` und `-o` anzugeben, in welche Richtung diese Regel greifen soll. Dies könnte dann so aussehen:

```
iptables -A FORWARD -m state --state NEW -p tcp --dport 80 -o eth2 -i eth0 -j ACCEPT
```

Listing 22.74 Firewallregel mit Angabe der Richtung über »-i« und »-o«

Bei der Regel aus Listing 22.74 wurde mit angegeben, dass die Regel nur für Pakete greift, die auf eth0 eintreffen und nach eth2 geroutet werden. Deutlich eleganter ist aber die Möglichkeit, den Paketfluss über selbst definierte Chains zu regulieren:

```
iptables -N int_to_ext
iptables -A FORWARD -i eth0 -o eth2 -j int_to_ext
```

Listing 22.75 Richtung über eine eigene Chain festlegen

Der erste Befehl aus Listing 22.75 erzeugt die neue Chain `int_to_ext`. Im zweiten Befehl wird eine Regel erzeugt, die alle Pakete aus der `FORWARD`-Chain, die über eth0 in das System gelangt sind und über eth2 das System verlassen sollen, an die neu angelegte Chain `int_to_ext` weitergibt.

In der neuen Chain können Sie alle Regeln setzen, die sich auf die entsprechende Traffic-Richtung beziehen (im Beispiel also auf den Datenverkehr vom LAN zum Internet). Sie können somit zum Beispiel den Webzugriff aus dem LAN erlauben, wie Sie in Listing 22.76 sehen können:

```
iptables -A int_to_ext -m state --state NEW -p tcp --dport 80 -j ACCEPT
```

Listing 22.76 Neue Regel in Bezug auf die eigene Chain

Trifft keine Regel aus der erstellten Chain auf ein Paket zu, so wird die Verarbeitung zurück an die aufrufende Chain gegeben – im Beispiel geht das Paket also zurück an die `FORWARD`-Chain.

Nun wollen wir ein Firewallskript erstellen, das eine Firewall mit drei Schnittstellen konfiguriert – je eine für extern, intern und für die DMZ. Folgende Bedingungen sollen erfüllt werden:



- ▶ Für die Firewall selbst wird nur der zwingend erforderliche Datenverkehr freigeschaltet. Dazu zählen die Überwachung per SNMP von einem Monitoring-Server aus, der Zeitabgleich mit einem externen Server und die Namensauflösung durch einen eigenen DNS-Server in der DMZ.

Damit die Firewall Updates für das Betriebssystem laden kann, wird der Zugriff auf einen Proxy-Server in der DMZ ebenfalls freigeschaltet. Zusätzlich wird zur Administration der SSH-Zugang von einer definierten Maschine aus dem internen Netz zugelassen.

- ▶ In der DMZ werden öffentlich zugängliche Dienste betrieben. Dazu zählen ein Webserver, ein Mailserver und ein DNS-Server.

- ▶ Die Systeme in der DMZ können alle auf einen externen NTP-Server zugreifen. Ansonsten sind nur die ausgehenden Ports geöffnet, die nötig sind, damit DNS und Mail korrekt funktionieren. Der Proxy-Server hat vollen Webzugriff nach außen. Die DMZ-Server können von einem Rechner des internen Netzes aus per SSH administriert werden.
- ▶ Die Clients im LAN dürfen über den Proxy-Server der DMZ in das Internet. Zusätzlich können sie per FTP Daten auf den Webserver der DMZ hochladen, den Monitoring-Server benutzen, Mails vom DMZ-Mailserver abrufen sowie alle Rechner der DMZ anpingen. Darüber hinaus dürfen die Clients natürlich auch all das, was externe Clients dürfen.
- ▶ Es wird ein minimaler Regelsatz für ICMP gesetzt.
- ▶ Zugriffe von der DMZ und dem externen Netz nach innen sind generell nicht erlaubt.

Den Netzwerkaufbau mit den entsprechenden Freigaben haben wir für Sie in Abbildung 22.4 noch einmal zusammengefasst.

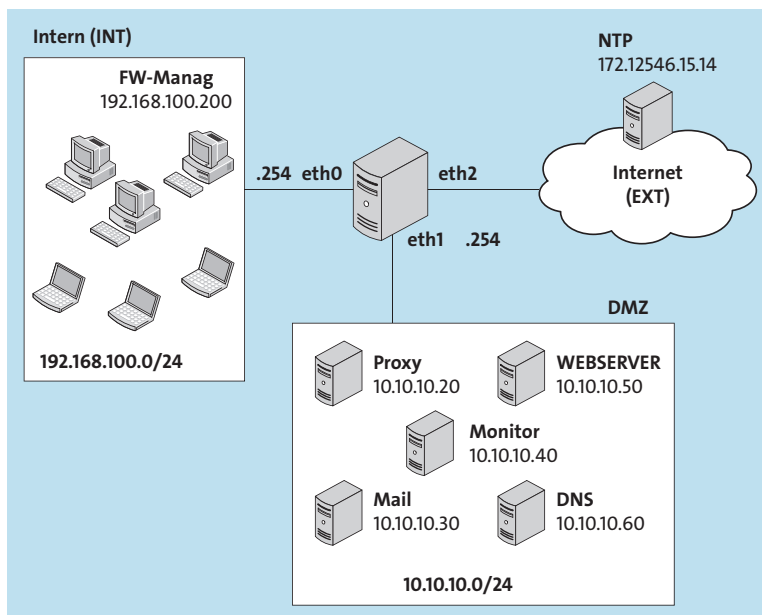


Abbildung 22.4 Netzwerkaufbau: Firewall mit drei Schnittstellen

Um die Anforderungen erfüllen zu können, müssen Sie ein Firewallskript mit dem Inhalt aus Listing 22.77 erstellen:

```
#!/bin/bash

# iptables binary
IPT="/sbin/iptables"
```

```
# Alte Konfiguration loeschen
$IPT -F
$IPT -X

# Default-Policy setzen
$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP

# Routing aktivieren
echo "1" > /proc/sys/net/ipv4/ip_forward

# Interfaces (Intern, DMZ, Extern)
INT=eth0
DMZ=eth1
EXT=eth2

# Management-Konsole fuer Firewall
MGMT=192.168.100.200

# Server DMZ
PROXY=10.10.10.20
MAIL=10.10.10.30
MONITOR=10.10.10.40
WEBSERVER=10.10.10.50
DNS=10.10.10.60

# Server extern
NTP=172.16.15.14

# Bestehende Verbindungen erlauben
$IPT -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Loopback freischalten
$IPT -A INPUT -i lo -j ACCEPT
$IPT -A OUTPUT -o lo -j ACCEPT

# Eigene Chains anlegen
$IPT -N int_to_dmz
$IPT -N dmz_to_ext
$IPT -N ext_to_dmz
```

```
# Verkehr aufteilen
$IPT -A FORWARD -i $INT -o $DMZ -j int_to_dmz
$IPT -A FORWARD -i $DMZ -o $EXT -j dmz_to_ext
$IPT -A FORWARD -i $EXT -o $DMZ -j ext_to_dmz

### Verkehr aus dem internen Netz in die DMZ
$IPT -A int_to_dmz -m state --state NEW -p tcp --dport 8080 -d $PROXY -j ACCEPT
$IPT -A int_to_dmz -m state --state NEW -p tcp --dport ftp -d $WEBSERVER -j ACCEPT
$IPT -A int_to_dmz -m state --state NEW -p tcp --dport https -d $MONITOR -j ACCEPT
$IPT -A int_to_dmz -m state --state NEW -p tcp -m multiport --dport pop3,imap \
    -d $MAIL -j ACCEPT
$IPT -A int_to_dmz -m state --state NEW -p tcp --dport ssh -s $MGMT -j ACCEPT
$IPT -A int_to_dmz -m state --state NEW -p icmp --icmp-type echo-request -j ACCEPT
$IPT -A int_to_dmz -j ext_to_dmz

### Verkehr aus der DMZ in das Internet
$IPT -A dmz_to_ext -m state --state NEW -p udp --dport ntp -d $NTP -j ACCEPT
$IPT -A dmz_to_ext -m state --state NEW -p tcp -m multiport --dport smtp,smtps \
    -s $MAIL -j ACCEPT
$IPT -A dmz_to_ext -m state --state NEW -p tcp -m multiport --dport http,https \
    -s $PROXY -j ACCEPT
$IPT -A dmz_to_ext -m state --state NEW -p tcp --dport domain -s $DNS -j ACCEPT
$IPT -A dmz_to_ext -m state --state NEW -p udp --dport domain -s $DNS -j ACCEPT
$IPT -A dmz_to_ext -j REJECT

### Verkehr aus dem Internet in die DMZ
$IPT -A ext_to_dmz -m state --state NEW -p tcp -m multiport \
    --dport smtp,smtps -d $MAIL -j ACCEPT
$IPT -A ext_to_dmz -m state --state NEW -p tcp -m multiport \
    --dport http,https -d $WEBSERVER -j ACCEPT
$IPT -A ext_to_dmz -m state --state NEW -p tcp --dport domain \
    -d $DNS -j ACCEPT
$IPT -A ext_to_dmz -m state --state NEW -p udp --dport domain \
    -d $DNS -j ACCEPT
$IPT -A ext_to_dmz -m state --state NEW -p icmp \
    --icmp-type fragmentation-needed -j ACCEPT
$IPT -A ext_to_dmz -j REJECT

### Regeln fuer die Firewall
# Eingehend
$IPT -A INPUT -i $INT -m state --state NEW -p tcp --dport ssh -s $MGMT -j ACCEPT
$IPT -A INPUT -i $DMZ -m state --state NEW -p udp --dport snmp -s $MONITOR -j ACCEPT
```

```
# Ausgehend
$IPT -A OUTPUT -o $DMZ -m state --state NEW -p tcp --dport 8080 -d $PROXY -j ACCEPT
$IPT -A OUTPUT -o $DMZ -m state --state NEW -p tcp --dport domain -d $DNS -j ACCEPT
$IPT -A OUTPUT -o $DMZ -m state --state NEW -p udp --dport domain -d $DNS -j ACCEPT
$IPT -A OUTPUT -o $EXT -m state --state NEW -p udp --dport ntp -j ACCEPT
$IPT -A OUTPUT -o $EXT -m state --state NEW -p icmp \
    --icmp-type fragmentation-needed -j ACCEPT
```

Listing 22.77 Firewallskript für eine Firewall mit drei Schnittstellen

In dem Skript aus Listing 22.77 werden nach dem Aufräumen, dem Erlauben von Antworten auf bestehende Verbindungen und dem Setzen von Variablen zunächst drei eigene Chains definiert: `int_to_dmz`, `dmz_to_ext` und `ext_to_dmz`. Damit diese den für sie zutreffenden Datenverkehr enthalten, werden unter Angabe der Optionen `-i` und `-o` Weiterleitungen in der FORWARD-Chain definiert. Anschließend folgen die Firewallregeln für die jeweiligen Chains.

Dabei sind besonders zwei Punkte wichtig. Zum einen sorgt der letzte Eintrag für die Regeln vom internen Netz zur DMZ, `$IPT -A int_to_dmz -j ext_to_dmz`, dafür, dass nach dem Durchlaufen von `int_to_dmz` noch die Chain `ext_to_dmz` nach passenden Regeln durchsucht wird. So ist gewährleistet, dass die internen Rechner zumindest genauso viel dürfen wie externe Besucher, ohne diese Regeln explizit einrichten zu müssen – dadurch vermeiden Sie doppelten Pflegeaufwand.

Der zweite wichtige Punkt ist die dedizierte Freigabe von ICMP. Es ist zwar möglich, ICMP vollständig zu unterbinden, dies kann allerdings zu unerwünschten Nebeneffekten führen. Erlauben Sie zumindest ICMP Typ 3 Code 4 *Fragmentation required* und *DF flag set*, damit die Automatismen zur Regulierung der Paketgrößen über die Firewallgrenzen hinweg funktionieren. Mit gesetztem `don't fragment-(DF)-Bit` im IPv4-Header darf ein Paket nicht fragmentiert, das heißt nicht in kleinere Pakete aufgeteilt werden. Sollte das aber nötig sein, weil ein Router in der Kommunikationskette mit einer kleineren *Maximum Transfer Unit* (MTU) arbeitet, dann wird das Paket verworfen, und ein *ICMP-Typ-3-Code-4-Paket* wird zurückgesendet. Wenn Ihre Firewall dieses Paket nun aufgrund eines zu strengen Regelwerks verwerfen sollte, kommt die Kommunikation unter Umständen gar nicht zustande.



Filtern Sie auch ausgehende Verbindungen!

Die Filterung von ausgehenden Verbindungen ist sehr wichtig und wird heutzutage leider immer noch sehr oft vergessen oder unterschätzt. Viele Administratoren gehen immer noch fälschlicherweise davon aus, dass ihre Systeme nie Ärger machen, korrumpiert werden oder einfach Fehlkonfigurationen aufweisen. Diese Annahmen sind nicht nur falsch, sondern auch gefährlich! Es besteht nicht nur die Gefahr, dass Ihre Infrastruktur als Basis für Angriffe genutzt wird, sondern Sie machen sich auch leicht zum Opfer. Ein beträchtlicher Teil der zurzeit



genutzten Angriffe lässt sich auf Schwachstellen von Webservern oder darauf laufenden Applikationen wie CMS-Systemen (Blogs, Webseiten etc.) zurückführen. Die so eingeschleuste Schadsoftware versucht zum einen, Kontakt nach Hause aufzubauen, um Schadcode nachladen zu können, und zum anderen versucht sie, gefundene Daten zu versenden oder sich über ein Botnetz fernsteuern zu lassen. Gerade Letztgenanntes lässt sich über strikt gesetzte Regeln zur Kommunikation nach außen erschweren oder sogar gänzlich verhindern.

22.6.9 Logging

Eines der Kernstücke einer Firewall ist das Logging. Dies hilft Ihnen nicht nur dabei, nachzuvollziehen, was passiert, sondern unterstützt Sie auch darin, herauszufinden, weshalb diese oder jene Kommunikation funktioniert oder nicht. Bei `iptables` wird zur Protokollierung das Target `LOG` verwendet. Da es sich dabei um ein nicht terminierendes Target handelt, werden Pakete nach dem Durchlauf zurück in die Verarbeitung gegeben. Alle Pakete, die `LOG` durchlaufen, werden an `syslog` übergeben.

In Listing 22.78 sehen Sie ein Beispiel für eine Regel, die über das Target `LOG` Meldungen an `syslog` weitergibt. Eine solche Log-Meldung aus der Datei `/var/log/syslog` wird anschließend mittels `tail` ausgegeben. Die Datei `/var/log/syslog` ist die Standard-Log-Datei bei Debian und Ubuntu, auf einem SUSE-System finden Sie die Meldungen in der Datei `/var/log/messages`.

```
root@saturn:~# iptables -A OUTPUT -m state --state NEW -p tcp --dport 80 -j LOG
root@saturn:~# tail -1 /var/log/syslog
Oct  9 10:03:32 saturn kernel: [560612.596981] IN= OUT=eth0 SRC=192.168.0.163 DST=\
    46.235.24.168 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=51197 DF PROTO=TCP \
    SPT=39430 DPT=80 WINDOW=5840 RES=0x00 SYN URGP=0
```

Listing 22.78 Anzeige in der Datei `»/var/log/syslog«` bei Verwendung des Targets `»LOG«`

Um Log-Meldungen besser unterscheiden zu können, können Sie die Option `--log-prefix` verwenden. Diese Option erwartet einen String, der jeder Log-Meldung vorangestellt wird. Listing 22.79 zeigt das Ergebnis bei der Verwendung von `--log-prefix`:

```
root@saturn:~# iptables -A INPUT -m state --state NEW -p tcp --dport ssh \
    -j LOG --log-prefix "SSH-Eingehend: "
root@saturn:~# tail -1 /var/log/syslog
Oct  9 10:33:49 saturn kernel: [562430.358969] SSH-Eingehend: IN=eth0 OUT=\
    MAC=00:26:18:81:1d:63:00:0a:e4:26:f4:3c:08:00 SRC=192.168.0.20\
    DST=192.168.2.10 LEN=60 TOS=0x00\
    PREC=0x00 TTL=64 ID=56235 DF PROTO=TCP\
    SPT=45175 DPT=22 WINDOW=5840 RES=0x00 SYN URGP=0
```

Listing 22.79 Log-Meldung mit Präfix

Damit können Sie Regeln sehr einfach im Log finden, Gruppen von Log-Einträgen bilden und diese zum Beispiel über Ihren Syslog-Server mittels Syslog-Regeln in eigene Dateien schreiben lassen.

Neugierig geworden?

Mehr zum Thema `syslog` finden Sie in Kapitel 12, »Syslog«.



Zunächst stellt sich die Frage: Was sollte protokolliert werden? Die einfachste, wenn auch unbefriedigende Antwort ist: alles, was Sie zur Fehlersuche oder zum Finden von Sicherheitsproblemen benötigen (oder was gegebenenfalls durch die Beauftragten für Datenschutz und IT-Sicherheit aus Ihrem Unternehmen vorgegeben wurde). Dies ist natürlich im Vorfeld schwer festzustellen. Daher ist es gang und gäbe, einfach jeden unterbundenen Datenverkehr zu protokollieren. Dies ist prinzipiell nicht falsch, benötigt aber viel Speicherplatz und eine leistungsfähige Firewall – Sie werden erstaunt sein, wie schnell sich ein Firewall-Log füllen kann. Diese Protokolle sind für Sie aber auch nur dann nützlich, wenn Sie sie auswerten und die richtigen Schlüsse daraus ziehen können. Wenn Sie hingegen tagelang mehrere Gigabyte große Log-Daten durchgehen, nur um festzustellen, dass der Angriff von einem infizierten PC eines brasilianischen Internetcafés stammte, dann hilft Ihnen das Logging höchstwahrscheinlich kein Stück weiter. Legen Sie den Fokus stattdessen auf unerlaubte ausgehende Verbindungen.

Die Anzahl der ausgehenden Verbindungen sollte in einem gut gepflegten Netzwerk sehr übersichtlich sein. Eine Untersuchung der Meldungen lohnt sich daher auf jeden Fall. Eine weitere Möglichkeit ist, den Datenverkehr nur ausschnittsweise zu protokollieren. Das ist mit dem *limit*-Modul möglich. Mit diesem Modul können Sie `iptables` dazu bringen, auf eine bestimmte Anzahl von Paketen pro Zeiteinheit zu achten und nur diese zu protokollieren. In Listing 22.80 sehen Sie, wie Sie das Modul *limit* verwenden können:

```
root@saturn:~# iptables -A INPUT -m state --state NEW -m limit \
    --limit 10/minute -p tcp --dport ssh -j LOG\
    --log-prefix "SSH-Eingehend: "
```

Listing 22.80 Verwendung von »limit« zur Reduzierung der Log-Einträge

Eine so wie in Listing 22.80 mit dem Modul *limit* versehene Regel protokolliert maximal zehn Pakete pro Minute. Das Modul *limit* kann auch noch mit anderen Intervallen umgehen: `second`, `hour` oder `day`. Darüber hinaus können Sie mit der Angabe von `--limit-burst` definieren, wie viele Pakete am Stück protokolliert werden sollen, bevor eine Pause gemacht werden soll. Damit können Sie sicherstellen, dass Ihr gesetztes Limit nicht bereits nach einem Sekundenbruchteil »aufgebraucht« ist. Ohne Angabe der Option `--limit-burst` liegt der Standardwert bei 5.



Verwenden Sie »limit« nicht, um »DoS«-Angriffe abzuwehren!

Wenn Sie das Modul *limit* auf eine andere Chain als die LOG-Chain anwenden, können Sie darüber auch die maximale Anzahl der Pakete für ein Zeitintervall definieren – was aber nicht sinnvoll und ratsam ist, da das *limit*-Modul nicht dazu geeignet ist, Datenverkehr sinnvoll einzuschränken.

Auch wenn oft die Empfehlung ausgesprochen wird, für verschiedene Dienste ein Limit zu setzen, um *Denial of Service*-(DoS-)Angriffe zu unterbinden, ist dies nicht sinnvoll. Um DoS-Angriffe abzuwehren, sollten Sie besser das Modul *connlimit* verwenden, das wir in Abschnitt 22.6.11, »Weitere nützliche Module für »iptables««, näher betrachten.

22.6.10 Network Address Translation und Masquerading

Als *Network Address Translation* (NAT) wird die Übersetzung einer IP-Adresse in eine andere bezeichnet. Da private Adressbereiche nicht im Internet geroutet werden, ist dies der normale Weg, um zum Beispiel aus dem heimischen Netzwerk über den DSL-Router ins Internet kommunizieren zu können, da somit nur der DSL-Router eine offizielle IP-Adresse benötigt. Dabei merkt sich der Router, welche interne IP-Adresse sich mit welchem externen Dienst verbinden will. Zu diesem Zweck betrachtet der Router die Quell- und Ziel-IP-Adresse und die Quell- und Zielports. Als Absenderadresse tauscht der Router dann die eigentliche Quell-IP-Adresse durch seine offizielle IP-Adresse aus. Bei den Antwortpaketen auf dieser Verbindung tauscht der Router die Adressen wieder zurück (also die offizielle IP-Adresse zurück in eine interne).

Diese Variante wird als *Source-NAT* oder kurz *SNAT* bezeichnet, da die Quell-IP-Adresse ausgetauscht wird. Der andere Weg, also der Austausch der Zieladresse, wird als *Destination-NAT* (DNAT) bezeichnet. Dies findet Anwendung, wenn Sie zum Beispiel einen Webserver mit einer internen IP-Adresse betreiben. Der Router muss Anfragen an seine offizielle IP-Adresse auf dem Port *tcp/80* entsprechend in die interne IP-Adresse des Webserver übersetzen. Zum Umsetzen von NAT gibt es bei *iptables* drei Targets: SNAT, DNAT und MASQUERADE.

Quelladressumsetzung mit SNAT

Die Regel aus Listing 22.81 übersetzt alle ausgehenden Pakete von 192.168.0.25 in die offizielle IP-Adresse 172.16.2.17:

```
root@saturn:~# iptables -A POSTROUTING -t nat -s 192.168.0.25 \
-j SNAT --to-source 172.16.2.17
```

Listing 22.81 Beispiel für die Umleitung mit »SNAT«

Nach der Angabe des Targets SNAT folgt mit der Option *--to-source* die Angabe der Quelladresse, auf die die Übersetzung erfolgen soll.

Dabei können Sie auch weitere Einschränkungen vornehmen. So können Sie ein *SNAT* auch für nur einen Port oder ein Protokoll einrichten. Auch die Einschränkung auf einen Portbereich, in den übersetzt werden soll, kann angegeben werden. Die folgende Regel übersetzt den ausgehenden Webverkehr des Netzes 192.168.100.0/24 in eine feste IP-Adresse. Der dafür genutzte Portbereich ist 15.000 bis 20.000:

```
root@saturn:~# iptables -A POSTROUTING -t nat -s 192.168.100.0/24 -p tcp \
    --dport http -j SNAT --to-source 172.16.2.25:15000-20000
```

Listing 22.82 Einschränkung des Portbereichs bei »SNAT«

Zieladressumsetzung mit DNAT

Das *DNAT* funktioniert analog zum *SNAT*. Um beispielsweise eingehende Pakete auf Port 80 an einen internen Webserver weiterzuleiten, genügt die Regel aus Listing 22.83:

```
root@saturn:~# iptables -A PREROUTING -t nat -p tcp --dport http \
    -j DNAT --to-destination 192.168.100.20
```

Listing 22.83 Beispiel für die Umleitung mit »DNAT«

Auch das Target *DNAT* erwartet eine Option, hier die Option `--to-destination`. Diese gibt die Zieladresse an, in die übersetzt werden soll.

Quelladressumsetzung mit MASQUERADE

Das *MASQUERADE* ist eine spezielle Variante von *SNAT*. Beim *SNAT* muss die Quelladresse, in die übersetzt wird, fest angegeben werden. Damit Sie *SNAT* auch in Umgebungen nutzen können, in denen die externe Adresse dynamisch ist (wie zum Beispiel beim heimischen DSL, bei dem sich meist alle 24 Stunden die offizielle IP-Adresse ändert), wurde das *MASQUERADE*-Target entwickelt. Listing 22.84 zeigt eine typische *MASQUERADE*-Regel:

```
root@saturn:~# iptables -A POSTROUTING -t nat -o eth1 -j MASQUERADE
```

Listing 22.84 Beispiel für die Umleitung mit »MASQUERADE«

Wie Sie Listing 22.84 entnehmen können, wurde bei der Regel keine feste Quelladresse angegeben, sondern nur das ausgehende Interface mit der Option `-o`. Mit dieser Konfiguration verwendet *iptables* als Quelladresse immer die gerade aktuelle Adresse des angegebenen Interface.

22.6.11 Weitere nützliche Module für iptables

iptables verfügt über viele weitere Module. In diesem Abschnitt wollen wir Ihnen noch zwei Module vorstellen, die Sie zur Steigerung der Sicherheit verwenden können: *connlimit* und *recent*.

Verbindungen limitieren: `connlimit`

Das Modul `connlimit` ermöglicht es Ihnen, die Anzahl der parallelen Verbindungen pro Client oder Netzwerk zu prüfen. Dies kann hilfreich sein, um zum Beispiel sicherzustellen, dass nicht ein Client alle Ressourcen verbraucht. Das Modul `connlimit` kann zwei Optionen verarbeiten:

1. `--connlimit-above <N>`

Mit dieser Option treffen nur Regeln zu, wenn mehr als `<N>` Verbindungen bestehen.

2. `--connlimit-mask <PREFIXLENGTH>`

Mit dieser Option können Sie angeben, dass sich das Limit immer auf ein Netzwerk bezieht und nicht auf die Gesamtzahl der Verbindungen.

Um zum Beispiel die gleichzeitigen Verbindungen aus einem `24er`-Netz auf Ihren Webserver zu beschränken, können Sie die Regel aus Listing 22.85 verwenden:

```
root@saturn:~# iptables -A FORWARD -d 172.16.1.1 -m state --state NEW \
    -m connlimit -p tcp --dport http --connlimit-above 10 \
    --connlimit-mask 24 -j REJECT
```

Listing 22.85 Limitierung der Verbindungsanfragen mit »`connlimit`«



Ungewollte Sperrungen mit »`connlimit`«

Eine Verbindungslimitierung ist ein zweischneidiges Schwert. Auf der einen Seite kann sie Angriffsversuche eindämmen oder defekte Clients zähmen, aber auf der anderen Seite können Sie auch Ihre Besucher aussperren. Gerade bei Anwendern hinter einem Proxy-Server kommt es oft zu ungewollten Sperrungen. Da der Proxy als Stellvertreter arbeitet, kommen die Anfragen aller Clients von der gleichen IP-Adresse. Daher können Sie nicht unterscheiden, wer genau die Verbindungen erzeugt. Besonders bei *LTE* gibt es bei den meisten Providern Zwangsproxys. Überlegen Sie daher genau, wie hart Sie die Regeln formulieren. Am besten ist es, wenn Sie die Pakete oberhalb des Limits nicht gleich verwerfen, sondern zunächst nur protokollieren. Damit können Sie ein Gefühl dafür bekommen, wie häufig Ihr Limit überschritten wird. Anschließend können Sie Obergrenzen definieren, um Ihre Umgebung vor Überlastung zu schützen.

Dynamische Sperrlisten: `recent`


Das Modul `recent` ist in der Lage, Absenderadressen einer zutreffenden Regel in eine dynamische Liste zu laden. Diese Liste können Sie wiederum in anderen Regeln verwenden und weiterverarbeiten. Damit können Sie relativ leicht ein kleines *Intrusion Prevention System* (IPS) aufbauen, das alle Zugriffe von Systemen unterbindet, die unerlaubt versucht haben, einmal (oder mehrfach) mit Ihrem Netzwerk zu kommunizieren. Möchten Sie zum Beispiel eine *Blacklist* erstellen, die alle Systeme enthält, die versuchen, auf Ihre DMZ mit dem Port

445 (Windows-Dateifreigaben) zuzugreifen, und die dann den Systemen auf dieser Liste jeglichen Zugriff in Ihr Netzwerk untersagt, so müssen Sie die Befehle aus Listing 22.86 absetzen:

```
iptables -A ext_to_dmz -m recent --update --name blacklist --seconds 60 -j DROP
iptables -A ext_to_dmz -p tcp --dport 445 -m recent --name blacklist --set
```

Listing 22.86 Erstellen einer Blacklist mit dem Modul »recent«


Der erste Befehl aus Listing 22.86 prüft, ob eine IP-Adresse auf der *Blacklist* ist und innerhalb der letzten 60 Sekunden aktiv war (`--seconds 60`). Falls dem so ist, wird das Paket verworfen (DROP) und der Eintrag aktualisiert (`--update`), sodass der Timer erneut auf 60 Sekunden gestellt wird. Der zweite Befehl befüllt die Liste. Jedes TCP-Paket an den Zielport 445 wird auf die *Blacklist* gesetzt.

Der Port 445 steht hier stellvertretend für beliebige Dienste, die Sie gar nicht anbieten. Wenn Sie lediglich Web- und Mailedienste anbieten, ist davon auszugehen, dass es sich bei dem Zugriffsversuch auf den Port 445 um einen Scan nach offenen Freigaben handelt und somit nicht um einen gültigen Kommunikationsversuch. 

Die Liste der aktuell gesperrten Adressen können Sie unter `/proc/net/xt_recent/<LISTNAME>` einsehen. Listing 22.87 zeigt ein Beispiel für eine solche Liste:

```
root@saturn:~# cat /proc/net/xt_recent/blacklist
src=192.168.0.20 ttl: 64 last_seen: 4354078769 oldest_pkt: 7 4354073600,\
  4354073900, 4354074500, 4354075699, 4354077869, 4354078169, 4354078769
```

Listing 22.87 Auszug aus der »Blacklist«

Im Übrigen werden die Einträge nicht automatisch gelöscht. Wenn die Liste voll ist, wird der älteste Eintrag überschrieben. Standardmäßig fasst die Liste 100 Einträge, wobei Sie beim Laden des Moduls jedoch einen anderen Wert angeben können. Falls Sie mit 1.000 Einträgen arbeiten möchten, können Sie den Befehl aus Listing 22.88 verwenden: 

```
root@saturn:~# modprobe xt_recent ip_list_tot=1000
```

Listing 22.88 Die maximale Anzahl von Einträgen in einer Liste erhöhen

Auch das Modul »recent« hat seine Tücken!

Vorsicht ist geboten, da Sie sich mit dem Modul *recent* ins eigene Fleisch schneiden können: Mit dem automatisierten Sperren können Sie sich schnell selbst ausschließen. Auch der Ausschluss von Unbeteiligten ist möglich, wenn zum Beispiel ein Angreifer mit gefälschten Adressen agiert. Damit können Sie unter Umständen schnell gültige und auch gewollte Kommunikation unterbinden. Verwenden Sie das Modul daher mit Bedacht, und überlegen Sie im Voraus, welche Konsequenzen auf Sie zukommen können!

22.6.12 Abschlussbemerkung

Mit *iptables* lassen sich hervorragend Paketfilter-Firewalls aufsetzen. Dank der unzähligen Module können auch die komplexesten Setups und Anforderungen umgesetzt werden – dabei bleiben nicht viele Wünsche offen. Auch etliche kommerzielle Tools oder Appliances nutzen unter der Haube *iptables*. Das Tool hat sich im Praxiseinsatz millionenfach bewährt und kann bedenkenlos eingesetzt werden.

Oft wird gegen die Nutzung von *iptables* angeführt, dass es komplex, unübersichtlich und aufwendig sei. Diese Feststellung gilt aber für alle Firewalls. Auch Lösungen mit bedienerfreundlichen GUIs führen im Hintergrund komplexe Berechnungen aus und täuschen die Einfachheit durch die grafische Oberfläche nur vor. Als Firewalladministrator müssen Sie nicht nur die Umgebung, sondern auch die Applikationen, deren Arbeitsweise und Kommunikationsabläufe sehr gut kennen. Ohne dieses Wissen bringt eine GUI Sie nur schneller ans falsche Ziel. Es gilt wie so oft der Grundsatz: »Sie sollten wissen, was Sie tun!«

Aber auch für *iptables* gibt es gleich mehrere grafische Oberflächen. Eine der bekanntesten und gleichzeitig sehr gut umgesetzten ist die *Shorewall*. Die mit *Shorewall* erstellten Skripte sind gut lesbar und umfangreich dokumentiert. So können Sie leicht überprüfen, ob die Intention einer Regel auch korrekt umgesetzt wurde. Die *Shorewall* finden Sie unter <https://shorewall.org/>. Alternativ können Sie sich auch *FireHOL* (siehe <https://firehol.org/>) oder das noch in der Entwicklung steckende (aber vielversprechende) *EasyWall* (siehe <https://github.com/jpylypiw/easywall>) ansehen.

22.7 DHCP

Das *Dynamic Host Configuration Protocol* (DHCP) dient zur automatischen Netzwerkkonfiguration von Systemen. Das Protokoll kann Systeme automatisiert mit IP-Adressen, sogenannten *Default Gateways*, einer Liste von DNS-Servern und einem *Fully Qualified Domain Name* (FQDN) versorgen. Die Spezifikation des DHCP-Standards in RFC2132 sieht über 60 zusätzliche Optionen vor. Von der Vergabe von NTP-Servern bis hin zu einer Auflistung des nächsten Druck- oder Mailservers ist alles möglich. Auch herstellerspezifische Optionen stehen zur Verfügung. Leider werden nicht von jedem DHCP-Client alle Optionen unterstützt. Prüfen Sie daher vorab, ob Ihre Umgebung die zusätzlichen Optionen unterstützt. Die grundsätzliche Netzwerkkonfiguration funktioniert aber immer. Mit DHCP ist es möglich, die Netzwerkkonfiguration für eine große Menge von Systemen zentral zu pflegen.

22.7.1 Funktionsweise

DHCP arbeitet als Client/Server-Dienst. Dabei senden Clients Anfragen in ein Netzwerk, die vom Server mit der passenden Konfiguration des Netzwerks beantwortet werden. Nach er-

folgreichem Austausch setzt der DHCP-Client die entsprechende Konfiguration dann entsprechend lokal um.

Die Clients senden ihre Anfragen, da der DHCP-Server ja nicht bekannt ist, als Broadcast von der Adresse 0.0.0.0 an 255.255.255.255. Die Anfrage wird als UDP-Paket auf den Port 67 gesendet und als *DHCPDISCOVER* bezeichnet. Der (oder die) DHCP-Server antwortet (bzw. antworten) auf diese Anfrage mit einem *DHCPOFFER*.

Dieses Paket enthält einen Vorschlag für eine IP-Adresse und wird vom Server auch an alle Systeme im Netzwerk gesendet. Der Client entscheidet sich dann für das aus seiner Sicht beste Angebot und fragt gezielt bei dem DHCP-Server mit einem *DHCPREQUEST* nach der ihm angebotenen IP-Adresse. Dies wird vom DHCP-Server in der Regel mit einem *DHCPACK* beantwortet. Daraufhin setzt der Client die angegebene Netzwerkkonfiguration um.

Die initiale Kommunikation über Broadcasts impliziert natürlich, dass DHCP immer nur in der aktuellen *Broadcast-Domain* funktioniert. Wenn Sie also mehrere Netzwerke mit DHCP betreiben wollen, müssen Sie den Server entweder auf dem zentralen Router betreiben oder einen sogenannten *DHCP-Relay-Agent* installieren. Dieser leitet die Anfragen aus einem Netz gezielt an einen DHCP-Server weiter.



22.7.2 Konfiguration

Einer der bekanntesten und weitverbreitetsten DHCP-Server ist der des *Internet Software Consortium* (ISC). Dieser Server wird in (fast) allen Distributionen als Standardserver verwendet. Je nach Distribution heißt das Paket entweder *dhcp*, *dhcp-server* oder *isc-dhcp-server*. Die Konfiguration erfolgt in der Datei *dhcpd.conf*, die Sie entweder unter */etc* oder unter */etc/dhcp* finden.

In der hierarchisch gegliederten Konfigurationsdatei können die meisten Optionen entweder global (für alle Clients), für einzelne Subnetze, für Adresspools oder individuell für einzelne Hosts gesetzt werden.

Dabei werden die DHCP-Optionen jeweils mit *option* eingeleitet. Danach folgen die zu setzende Option und der entsprechende Wert dafür. Nachstehend haben wir für Sie die wichtigsten Optionen aufgelistet:

- ▶ *domain-name*
setzt den Domainnamen des Rechners.
- ▶ *domain-name-servers*
Liste der für dieses Netz gültigen DNS-Server
- ▶ *domain-search*
gibt die Domainsuchliste an.

▶ routers

Hier können Sie eine Liste mit Routern in Ihrem Netz angeben – in der Regel wird hier nur das Default-Gateway angegeben. Sie können jedoch auch mehrere Router angeben.

▶ broadcast-address

Diese Option setzt die Broadcast-Adresse für die Clients.

Jede per DHCP verteilte Netzwerkkonfiguration hat eine Gültigkeitsdauer, die sogenannte *Lease Time*. Diese müssen Sie auch definieren. Dabei wird die *Lease Time* von zwei Direktiven gesteuert: *default-lease-time* und *max-lease-time*. Beide Direktiven erwarten als Argument eine Zeitangabe in Sekunden. Die *default-lease-time* bestimmt die normale Dauer des Leases. Wenn ein Client keinen Wunsch bezüglich der Lease-Dauer äußert, wird der hier angegebene Wert verwendet. Mit *max-lease-time* wird die maximal gültige Dauer definiert.

Jedes Netzwerk, für das Ihr DHCP-Server IP-Adressen vergeben soll, muss über einen eigenen subnet-Abschnitt verfügen. Darin definieren Sie die Direktive *range*, mit der Sie den Bereich spezifizieren, aus dem der DHCP-Server IP-Adressen verteilen darf – dieser Bereich wird auch als *Pool* bezeichnet.



Listing 22.89 zeigt die Konfiguration eines DHCP-Servers für die Domäne *example.net*. Dort läuft sowohl der DNS-Server als auch das Default-Gateway auf 192.168.2.1. Die Gültigkeitsdauer von DHCP-Einträgen wird auf 600 Sekunden (10 Minuten) und maximal 7200 Sekunden (2 Stunden) gesetzt. Dem DHCP-Server wurde zur Adressvergabe der Bereich von 192.168.2.150 bis 192.168.2.250 zugeteilt.

```
option domain-name "example.net";
option domain-name-servers 192.168.2.1;
option routers 192.168.2.1;
default-lease-time 600;
max-lease-time 7200;
subnet 192.168.2.0 netmask 255.255.255.0 {
    range 192.168.2.150 192.168.2.250;
}
```

Listing 22.89 Beispielkonfiguration eines DHCP-Servers

Damit Ihr DHCP-Server mehrere Netze versorgen kann, müssen Sie lediglich die *subnet*-Direktive wiederholen und gegebenenfalls subnetzspezifische DHCP-Optionen aus der globalen Sektion in das Subnetz übernehmen. Listing 22.90 zeigt eine Beispielkonfiguration für die Subnetze 192.168.2.0/24 und 192.168.100.0/24:

```
option domain-name "example.net";

default-lease-time 600;
max-lease-time 7200;
```

```

subnet 192.168.2.0 netmask 255.255.255.0 {
    range 192.168.2.150 192.168.2.250;
    option domain-name-servers 192.168.2.1, 192.168.100.1;
    option routers 192.168.2.1;
}

subnet 192.168.100.0 netmask 255.255.255.0 {
    range 192.168.100.100 192.168.100.150;
    option domain-name-servers 192.168.100.1, 192.168.2.1;
    option routers 192.168.100.1;
}

```

Listing 22.90 Beispielkonfiguration mit zwei Subnetzen

Wie Sie Listing 22.90 entnehmen können, wurden die zuvor in Listing 22.89 global gesetzten Direktiven `domain-name-servers` und `routers` nun in die jeweiligen `subnet`-Abschnitte aufgenommen. Der DHCP-Server pflegt auch eine Liste von vergebenen Leases. Die Liste wird in der Datei `dhcpd.leases` vorgehalten. Je nach Distribution finden Sie diese Datei unter `/var/lib/dhcp3` oder `/var/lib/dhcp`. In Listing 22.91 sehen Sie eine Liste aller vergebenen IP-Adressen:

```

root@debian:~# cat /var/lib/dhcp/dhcpd.leases
lease 192.168.2.150 {
    starts 1 2023/01/02 08:38:10;
    ends 1 2023/01/02 08:48:10;
    cltt 1 2023/01/02 08:38:10;
    binding state active;
    next binding state free;
    rewind binding state free;
    hardware ethernet 08:00:27:fe:67:04;
    uid "\377'\376g\004\000\001\000\001+ETq\010\000'\376g\004";
    client-hostname "ubuntu";
}

```

Listing 22.91 Liste aller vergebenen IP-Adressen

Wie Sie Listing 22.91 entnehmen können, wird auch der Inhalt der Datei `dhcpd.leases` hierarchisch aufgebaut – pro Eintrag finden Sie einen mit geschweiften Klammern umschlossenen Block, der mit dem Schlagwort `lease` <IP-ADRESSE> eingeleitet wird.

Falls Sie einzelne Systeme separat konfigurieren möchten, müssen Sie nicht ein 32er-Subnetz konfigurieren. Dafür können Sie in dem vorhandenen Subnetz einfach ein *Host-Objekt* anlegen. Darin können Sie durch Angabe der MAC-Adresse ein System spezifizieren und dafür eine gesonderte Konfiguration vornehmen. So können Sie zum Beispiel eine Reservierung mit der Option `fixed-address` vornehmen, sodass dieses System immer die gleiche

IP-Adresse zugewiesen bekommt. Optional können Sie mit `hostname` auch den Namen des Clients fest vergeben. In Listing 22.92 sehen Sie eine Subnetzdeklaration mit einer festen Adresszuordnung für `client1`:

```
subnet 192.168.2.0 netmask 255.255.255.0 {
    range 192.168.2.150 192.168.2.250;
    option domain-name-servers 192.168.2.1, 192.168.100.1;
    option routers 192.168.2.1;

    host client1 {
        hardware ethernet 00:0a:e4:26:f4:3c;
        fixed-address 192.168.2.101;
    }
}
```

Listing 22.92 Zuweisung einer reservierten IP-Adresse



Achten Sie auf eventuell vorhandene statische IP-Adressen!

Vergeben Sie statische IP-Adressen nicht aus dem mit `range` angegebenen dynamischen Bereich (Pool). Ansonsten laufen Sie Gefahr, dass eine IP-Adresse von zwei Systemen genutzt wird.

Kapitel 23

DNS-Server

DNS wird stets unterschätzt und doch immer gebraucht! In diesem Kapitel zeigen wir Ihnen, wie Sie Ihren eigenen Nameserver aufsetzen und betreiben können (als Rekursive-DNS oder als eigenen autoritativen DNS-Server, der selbst Zonen verwaltet), wie Sie es vermeiden, sich am Schwergewicht »DNSSEC« zu verheben, und wie Sie die verschlüsselte Variante »DNS over HTTPS« (DoH) einrichten.

Das *Domain Name System* (DNS) dient primär dazu, Namen in IP-Adressen und IP-Adressen in Namen aufzulösen. In den Anfangstagen des Internets, als es noch *ARPANET* hieß und nur eine Handvoll Teilnehmer hatte, wurde die Namensauflösung von jedem Administrator eines Netzbereichs in einer Datei per Hand vorgenommen. Alle Änderungen mussten dabei stets an die übrigen Teilnehmer versandt werden, damit diese in deren lokalen *hosts*-Dateien eingepflegt werden konnten. Ein ziemlicher Aufwand, der dringend nach einer anderen Lösung verlangte. Diese wurde im Jahre 1983 mit DNS geschaffen: einem hierarchischen, verteilten System zum Auflösen von Namen in IP-Adressen und von IP-Adressen in Namen.

23.1 Funktionsweise

Zur Auflösung von Namen fragt ein Client zunächst immer den lokalen *resolver*. Dieser weiß, über welche Wege Namen aufgelöst werden können. Dabei wird, auch heutzutage noch, zuerst in die lokale Hostdatei geschaut. Wird dort kein Eintrag gefunden, fragt der *resolver* einen der konfigurierten Nameserver. Auf der Serverseite kann eine DNS-Anfrage mit drei verschiedenen Verfahren beantwortet werden:

► **autoritativ**

Der gefragte Nameserver ist selbst für die Zone verantwortlich und holt die Daten aus einer lokalen Zonendatei.

► **nichtautoritativ**

Der gefragte Nameserver ist nicht selbst für die Zone verantwortlich und muss die Daten ermitteln. Dabei wird zwischen den folgenden Abfragearten unterschieden:

– *rekursiv*

Der Server holt die Daten von einem anderen Nameserver, arbeitet also als Proxy (Stellvertreter): Client-Server-Verfahren.

– *iterativ*

Der Server antwortet mit einem oder mehreren Verweisen oder einem *Resource Record* auf andere Nameserver, die den Namen auflösen können: Verfahren zwischen Nameservern.

Üblicherweise werden Clientanfragen rekursiv gestellt. Daher beginnt der Ablauf also damit, zu prüfen, wer für eine Zone zuständig ist. Dabei wird das Pferd von hinten aufgezäumt. Falls der gefragte DNS-Server nicht für die Zone selbst zuständig ist, fragt er die Root-Server (die die oberste Hierarchieebene im DNS darstellen), wer für die angefragte TLD (*Top Level Domain*) zuständig ist.

Ist diese Information bekannt, fragt er die Server der TLD-Ebene, wer für die Domäne verantwortlich ist. Anschließend wird der zuständige DNS-Server gefragt, wie der Name lautet, und die Antwort wird dem Client zurückgegeben.

Der Ablauf erfolgt also streng hierarchisch – von oben nach unten, vom Punkt zum Hostnamen. Dabei stellen Clients stets rekursive Anfragen. Im Hintergrund werden aber alle Verfahren genutzt, um eine Anfrage beantworten zu können (siehe Abbildung 23.1).

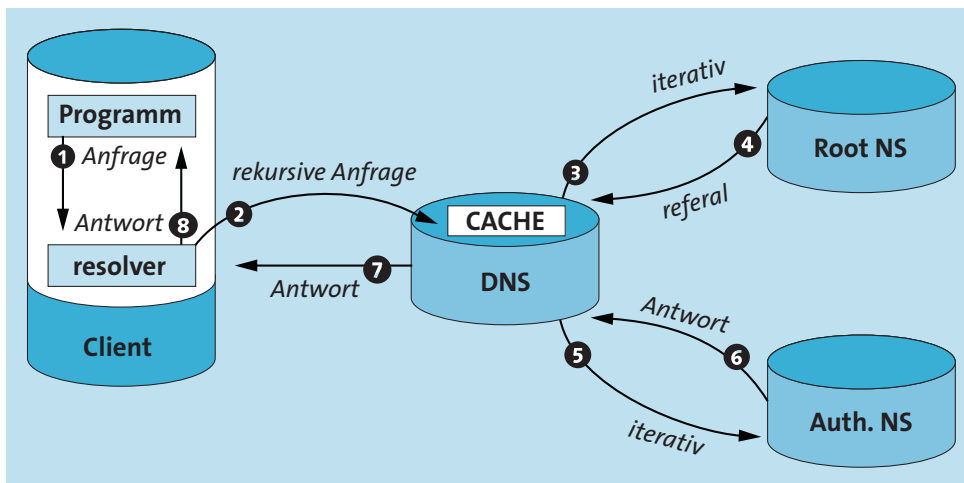


Abbildung 23.1 Ablauf einer Namensauflösung



Abschluss mit: ».«

DNS-Namen müssten eigentlich immer mit einem abschließenden Punkt angegeben werden, also zum Beispiel als "www.rheinwerk-verlag.de.", da dieser Punkt den Namen abschließt.

Der abschließende Punkt kann zwar weggelassen werden, ist aber streng genommen Bestandteil des Namens. Auf diese Besonderheit kommen wir im weiteren Verlauf des Kapitels noch zurück.

23.1.1 Unterschied: rekursiv und autoritativ

Bei DNS-Servern wird zwischen rekursiven und autoritativen Servern unterschieden. Ein rekursiver DNS ist »nur« ein Proxy-Server (man könnte auch »Stellvertreter« sagen). Er holt die angefragten Informationen ab und stellt das Ergebnis zur Verfügung. Von einem rekursiven Server erhalten Sie stets sogenannte *non authoritative*, also nicht autorisierte Antworten, da er selbst nicht für abgefragte Namen zuständig ist.

Ein autoritativer Server hingegen ist für eine (oder mehrere) Zonen zuständig und beantwortet nur Anfragen an Zonen, für die er selbst zuständig ist. Dafür ist die Rückmeldung aber autoritativ, also verbindlich.

23.1.2 Einträge im DNS: Resource Records

Namen sind nicht gleich Namen. Was auf den ersten Blick komisch anmutet, ist aber rein logisch zu verstehen. Damit die DNS-Server die Unterschiede in den jeweiligen Anfragen ausmachen können, wurden die sogenannten *Resource Records* (oder kurz *Records* oder *RR*) geschaffen. Anhand dieser Records kann ein DNS-Server unterscheiden, welcher Natur die Anfrage ist und wie diese beantwortet werden muss. In der nachstehenden Auflistung haben wir für Sie die gängigsten Record-Typen und deren Bedeutung aufgelistet:

- ▶ A
liefert eine IPv4-Adresse zu einem Namen zurück. Typische Antwort: 192.168.0.20
- ▶ AAAA
liefert eine IPv6-Adresse zu einem Namen zurück. Typische Antwort:
fe80::a00:27ff:fee9:eb75
- ▶ PTR
liefert einen Namen zu einer IPv4/IPv6-Adresse zurück – auch als *Reverse Record* bezeichnet. Typische Antwort: proxy.example.net.
- ▶ CNAME
stellt einen Verweis, eine Weiterleitung oder einen Alias dar.

Nicht auf Domänenebene

Beachten Sie, dass *CNAME*-Records nur auf Namen und nicht auf die Zone selbst verweisen dürfen. So darf der Name *www.example.net* als *CNAME* auf *example.net* verweisen, allerdings können Sie nicht *example.net* auf *www.example.net* verweisen lassen!



- ▶ MX
liefert den zuständigen Mailserver (*mail exchange*) für eine Zone zurück. Typische Antwort: 10 mail.example.net.

- ▶ NS
liefert den zuständigen Nameserver für eine Zone zurück. Typische Antwort:
`dns.example.net.`
- ▶ SRV
liefert einen Server zurück, der einen Dienst anbietet, der meist in einem Windows-Active Directory zu finden ist. Mögliche Antwort: `ldap.example.net.`
- ▶ TXT
liefert einen Text zurück. Mögliche Antwort: "Hello World"
- ▶ SOA
liefert einen Ansprechpartner und Parameter zur abgefragten Zone zurück (SOA: engl. für *Start of Authority*). Typische Antwort:
`dns.example.de. admin.example.net. 2021010401 28800 7200 604800 3600`

Dies sind bei Weitem nicht alle Record-Typen. Das *Domain Name System* hält noch viele Möglichkeiten bereit, die ein normalsterblicher Nutzer selten wahrnimmt, die für den Betrieb von Diensten (wie MX-Records für Mail oder SRV-Records für ein Windows-AD) oder des Internets jedoch unerlässlich sind. Jeder Record-Typ hat eine unterschiedliche Anzahl an Werten, die er ausliefern kann. Liefern zum Beispiel A- oder PTR-Records nur die IP-Adressen oder Namen zurück, so wird bei MX-Records gleichzeitig noch eine Gewichtung in Form eines Zahlenwerts mit ausgeliefert.

Bei SRV-Records wird sogar noch die Portnummer des angebotenen Dienstes mit ausgeliefert. Im Laufe der Zeit ist das ursprünglich als einfaches Adressbuch gedachte DNS-System immer weiter gewachsen.

23.1.3 Die Grundkonfiguration

Der wohl bekannteste und weltweit am meisten genutzte Nameserver ist der *Berkeley Internet Name Daemon* (BIND). Die Software wird ähnlich wie der DHCP-Server vom ISC gepflegt und weiterentwickelt. BIND ist für so gut wie jede Plattform verfügbar. Je nach Distribution heißt das passende Paket *bind* oder *bind9*.

Nach der Installation aus den Paketquellen müssen Sie zunächst einige grundlegende Einstellungen vornehmen und festlegen, für welche *Zonen* (Domains) Sie verantwortlich sind. Diese Einstellungen werden in der *named.conf* vorgenommen.

Je nach Distribution finden Sie diese entweder direkt unter */etc* oder in dem Unterverzeichnis */etc/bind*. Bei einigen Distributionen (wie zum Beispiel Debian und Ubuntu) wird die Konfiguration auch auf verschiedene Dateien aufgeteilt, sodass in der Hauptkonfigurationsdatei lediglich die einzelnen Konfigurationsdateien inkludiert werden. Listing 23.1 zeigt den Ausschnitt der Datei *named.conf* von einem Debian- oder Ubuntu-System:

```
include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Listing 23.1 »Include«-Einträge in der »named.conf«

Im Folgenden wird stets mit der Datei *named.conf* gearbeitet. Der erste entscheidende Block in der Konfigurationsdatei ist der *options*-Block. Jeder Block wird durch geschweifte Klammern eingeleitet und beendet, wobei jede Zeile (auch das Ende eines Blocks) mit einem Semikolon abgeschlossen werden muss. Kommentare können durch *//* oder *#* eingeleitet werden. Ganze Kommentarblöcke können durch */** und **/* gekapselt werden. Listing 23.2 zeigt einen *options*-Block:

```
options {
    directory "/var/cache/bind";

    dnssec-validation auto
    listen-on-v6 { any; };
};
```

Listing 23.2 »options«-Block aus der »named.conf«

Die Direktive *directory* spezifiziert das Arbeitsverzeichnis des Nameservers. Dateien, die innerhalb der Konfigurationsdatei angegeben werden, beziehen sich immer relativ auf dieses Verzeichnis. Die Angabe von absoluten Pfaden ist aber auch möglich.

Unterschiedliche Pfade je Distribution

Bei SUSE-Systemen wird hier standardmäßig der Pfad */var/lib/named* verwendet, bei CentOS */var/named* und unter Debian und Ubuntu */var/cache/bind*. Um die Beispiele übersichtlicher zu gestalten, beziehen wir uns darin immer auf */var/cache/bind*.



Mit *listen-on-v6* kann bestimmt werden, auf welchen lokalen IPv6-Adressen der Nameserver lauschen soll. Im Beispiel aus Listing 23.2 wurde für *listen-on* die Option *any* verwendet – dadurch lauscht der Nameserver einfach auf allen IPv4- und IPv6-Interfaces des Servers. Durch die Angabe von *none* bei der Direktive *listen-on-v6* wird es abgeschaltet.

Um den Nameserver auf bestimmte Adressen zu beschränken, können Sie eine Liste von IPv6-Adressen angeben, auf denen der Nameserver lauschen soll – für IPv4-Adressen müssen Sie die Direktive *listen-on* verwenden. Selbstverständlich müssen die einzelnen Einträge jeweils mit einem Semikolon abgeschlossen werden. Auch die Angabe eines Ports ist möglich, was allerdings nur selten sinnvoll ist. Der Standardport ist 53 (TCP und UDP). In Listing 23.3 sehen Sie eine Mehrfachzuweisung von IP-Adressen und Ports:

```
listen-on port 10053 { 192.168.1.100; 192.168.2.100; };  
listen-on { 192.168.100.100; };
```

Listing 23.3 Angabe der Ports und IP-Adressen über »listen-on«

Um sich den DNS-Cache Ihres Providers zunutze zu machen, können Sie mit der Direktive `forwarders` benachbarte Nameserver angeben. Bei Anfragen, die nicht aus Ihrem Cache beantwortet werden können, werden diese dann befragt.

Darüber hinaus kann mit der Direktive `forward` festgelegt werden, ob Anfragen ausschließlich an die anderen Nameserver weitergeleitet werden sollen (`only`) oder ob Ihr Server beim Scheitern der Abfrage selbst eine rekursive Anfrage starten soll (`first`). In Listing 23.4 sehen Sie ein Beispiel für die `forwarders`-Direktive:

```
options {  
    [...]  
    forward first;  
    forwarders {  
        10.10.10.1;  
        10.10.20.1;  
    };  
    [...]  
};
```

Listing 23.4 Beispiel für die »forwarders«-Direktive

Der nächste entscheidende Block beinhaltet die Zonendefinitionen, die mit `zone` eingeleitet werden. Um z.B. die Domain *example.net* zu betreiben, müssen Sie folgende Zone definieren:

```
zone "example.net" {  
    type primary;  
    file "primary/example.net";  
};
```

Listing 23.5 Definition der Zone »example.net«

Die Direktive `type` gibt den Zonentyp an. In diesem Fall ist es `primary`, weil dieser Server der primäre DNS-Server der Domain *example.net* werden soll. Mit der Direktive `file` wird angegeben, in welcher Datei die Zonendaten für *example.net* liegen. Wie bereits erörtert wurde, ist der Pfad relativ zu dem mit `directory` gesetzten Pfad im `options`-Block (im Beispiel also `var/cache/bind/primary/example.net`).

23.1.4 Zonendefinitionen

Eine Zone enthält alle relevanten Informationen für eine Subdomain. Diese einzelnen Informationen werden als sogenannte *Resource Records* (RR) hinterlegt. Auch in Zonendateien

können Kommentare mit einem Semikolon eingeleitet werden. Vor dem ersten RR müssen Sie mit `$TTL` die Standard-*Time-To-Live* für alle RRs angeben. Dieser Wert gibt an, wie lange Einträge gültig sind, bevor diese neu abgefragt werden müssen. Die Angabe erfolgt dabei in Sekunden. Mit unterschiedlichen Suffixen können aber auch Wochen *w*, Tage *d*, Stunden *h* oder Minuten *m* angegeben werden.

Nach der TTL folgt der wichtigste Resource Record, der *SOA*. Das Kürzel steht für *Start Of Authority* und definiert die Zuständigkeit und die Eigenschaften der Zone. Abbildung 23.2 zeigt den Aufbau eines SOA-Records.

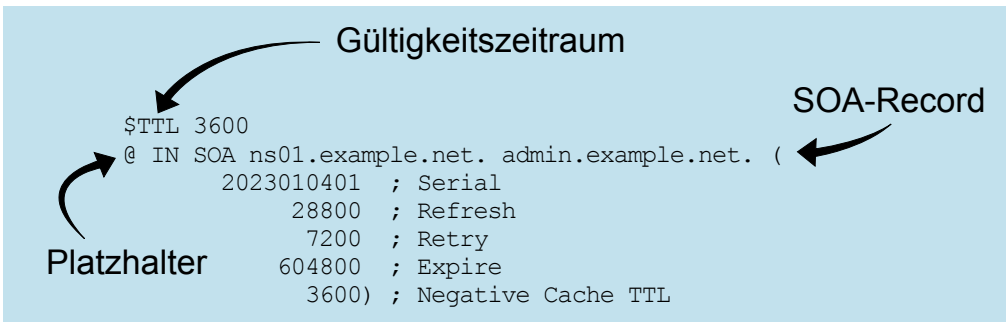


Abbildung 23.2 Aufbau des »SOA«-Records

Das `@`-Zeichen am Anfang der zweiten Zeile (zu Beginn des SOA-Records) gibt an, dass sich alle Einträge auf die in der `/etc/named.conf` definierte Domain beziehen. Die Schreibweise hat den Vorteil, dass von mehreren Zonendefinitionen auf die gleiche Zonendatei verwiesen werden kann. So können Sie zum Beispiel die Records der Domänen *example.de*, *example.net* und *example.com* in nur einer Datei vorhalten. In diesem Beispiel verweisen wir von *example.net* auf die Zonendatei. Ausgeschrieben sähe die erste Zeile des SOA-Records so aus:

```
example.net.      IN      SOA      ns1.example.net. admin.example.net. (
```

Listing 23.6 »SOA-Record« ohne die Verwendung des Platzhalters »@«

Mit `IN` wird angegeben, dass es sich bei dem folgenden RR um die Klasse *Internet* handelt. Nach der Angabe des Record-Typs mit `SOA` folgt der FQDN des primären Name-servers der Zone, in diesem Fall ist es `ns1.example.net.` (die Angabe muss mit einem Punkt abgeschlossen werden!). Danach folgt die E-Mail-Adresse des zuständigen Administrators.

Das `@`-Zeichen muss hier allerdings durch einen Punkt ersetzt werden – somit steht `admin.example.net` also für `admin@example.net`. Falls die E-Mail-Adresse vor dem `@`-Zeichen einen Punkt enthält, muss dieser mit einem Backslash maskiert werden. Die E-Mail-Adresse `max.mustermann@example.net` muss daher als `max\.mustermann@example.net` angegeben werden. Anschließend folgen die SOA-spezifischen Konfigurationen in Klammern. Dabei müssen folgende Angaben gemacht werden:

► Serial

Serial steht für die Versionsnummer der Zone. Bei jeder Änderung an der Zonendatei muss sie inkrementiert werden. Die Seriennummer wird von den Secondary Nameservern benutzt, um zu sehen, ob sie die Zone neu herunterladen müssen. Der Wertebereich für Serial beträgt 32 Bit. Im Prinzip könnten Sie bei 1 anfangen zu zählen. Es hat sich jedoch eingebürgert, Serial in folgendem Format darzustellen: YYYYMMDDXX. Dabei steht YYYY für die vierstellige Jahreszahl, MM für den aktuellen Monat, DD für den Tag des Monats und XX für einen fortlaufenden Zähler, der angibt, wie oft die Zone bereits an diesem Tag geändert wurde.

► Refresh

Mit Refresh wird angegeben, nach welcher Zeit die Secondary Nameserver beim Primary nachschauen sollen, um zu prüfen, ob die Serial sich geändert hat.

► Retry

Retry legt fest, nach welcher Zeit ein Secondary Nameserver erneut nachfragen soll, wenn der erste Versuch fehlgeschlagen ist.

► Expire

Mit Expire wird quasi das Haltbarkeitsdatum für die Zone der Secondary Nameserver angegeben. Sollte der Secondary Nameserver nach Ablauf dieser Zeit noch immer keinen Kontakt zum Primary herstellen können, so wird die Zone ungültig.

► Negative Cache TTL (früher *minimum*)

Gibt die Zeit in Sekunden an, die eine negative Rückmeldung des Servers im Cache vorgehalten werden darf.



Beachten Sie die Bedeutung von »Negative Cache TTL« (früher: »minimum«)

Die Negative Cache TTL (früher *minimum*) besaß im Laufe der Zeit unterschiedliche Bedeutungen. Sie werden daher noch in vielen Beispielen im Internet und in älteren Büchern eine andere Bedeutung finden. Ursprünglich gab der Wert an, wie lange die Standard-TTL der Resource Records dauert. Daher stammt auch der Name *minimum*. Durch RFC3208 wurde das aber abgelöst. Stattdessen wird heutzutage der \$TTL-Eintrag aus der ersten Zeile der Zonendatei verwendet.

Richtig gesetzte Werte entscheiden über Erfolg und Misserfolg Ihres DNS-Servers. Die DENIC zum Beispiel schreibt für *.de*-Domains folgende Wertebereiche vor:

- refresh: 3.600 – 86.400 Sekunden
- retry: 900 – 28.800 Sekunden
- expire: 604.800 – 3.600.000 Sekunden
- minimum (negTTL): 180 – 86.400

Auf www.denic.de/hintergrund/nameservice/nameserver-und-nsentry-eintraege.html finden Sie weitere Informationen zu den vorgeschriebenen Werten. Nach dem SOA-Record folgen weitere RR. Für eine gültige Zonendatei muss mindestens noch ein NS-Record definiert werden, der den verantwortlichen Nameserver für die Domain bestimmt. Der Resource Records sind dabei stets so aufgebaut:

```
<NAME> <TTL> <CLASS> <TYPE> <RDATA>
```

Listing 23.7 Aufbau eines »Resource Records«

Dabei haben die einzelnen Begriffe folgende Bedeutung:

- ▶ **<NAME>**
gibt den Namen für den RR an. Für einen Namen, zum Beispiel *www.example.net*, kann sowohl die Kurzform *www* als auch die Langform *www.example.net* verwendet werden. Bei der Langform ist der abschließende Punkt Pflicht – ohne diesen würde BIND den Zonennamen anfügen. Mit der Angabe ohne Punkt würde der Name also als *www.exmaple.net.example.net* erkannt werden. Wenn mehrere RR für den gleichen Namen definiert werden sollen, kann der Name weggelassen werden. Stattdessen muss die Zeile mit einem Leerzeichen beginnen – wir empfehlen Ihnen aber zur Wahrung der Übersicht, diese Funktion nicht zu verwenden.
- ▶ **<TTL>**
Die TTL gibt an, wie lange der Eintrag im Cache gehalten werden darf. Die Angabe der TTL ist optional. Wird keine TTL angegeben, so wird die Standard-TTL verwendet, die mit \$TTL in der ersten Zeile der Zonendatei angegeben wurde.
- ▶ **<CLASS>**
Hier wird der Klassentyp angegeben. Der Klassentyp muss nur einmal während der Zonendefinition angegeben werden. In der Regel wird IN für *Internet* verwendet. Andere mögliche Werte sind CH für *Chaosnet* oder HS für *Hesiod*, diese werden aber kaum noch benutzt und spielen eigentlich keine Rolle mehr.
- ▶ **<TYPE>**
Hier wird der Typ des RR angegeben. Die gebräuchlichsten Typen haben wir Ihnen bereits zu Beginn des Abschnitts vorgestellt. Diese werden wir gleich näher betrachten.
- ▶ **<RDATA>**
Unter <RDATA> werden die Daten des Resource Records abgelegt. In der Regel sind das eine IP-Adresse, ein Name oder etwas komplexere Angaben wie beim SOA-Record.

Schauen wir uns die gebräuchlichsten Records einmal genauer an:

▶ **A- und AAAA-Records**

Der am häufigsten verwendete Record ist der A-Record, da dieser für die Zuweisung von Hostnamen zu IPv4-Adressen verwendet wird. Ein A-Record wird wie folgt definiert:

```
ns1.example.net.      12h      IN      A      192.168.2.10
```

Für IPv6-Adressen steht der AAAA-Record zur Verfügung:

```
ns1.example.net.      12h      IN      AAAA    2001:db8::1
```

► NS-Records

Mit NS-Records werden die für eine Zone zuständigen Nameserver definiert. Aus Gründen der Redundanz sollten Sie mindestens zwei Nameserver in unterschiedlichen Netzen haben. Standardmäßig sieht ein NS-Record wie folgt aus:

```
example.net.  12h      IN      NS      ns1.example.net.
example.net.  12h      IN      NS      ns2.example.net.
```



Achtung: »Glue« nicht vergessen!

Befindet sich der Nameserver für eine Zone (wie in diesem Beispiel) selbst in der Zone, muss ein sogenannter *Glue-Record* erzeugt werden. Ein Glue-Record ist nichts weiter als ein *A-Record* für den Nameserver, der eine Ebene höher definiert wird. In diesem Fall wäre ein Glue-Record in der Zone `.net` notwendig. Ohne Glue-Record könnten Clients sich nicht an den für *example.net* verantwortlichen Nameserver wenden, da die IP-Adresse des Nameservers ja erst in der Zone definiert wird.

► MX-Records

Die Angabe der für eine Zone zuständigen Mailserver erfolgt mit dem *Mail Exchange* (MX)-Record. Ein MX-Record besteht immer aus einer Präferenz und dem Namen des Mailservers:

```
example.net.      12h      IN      MX      10      mx1.example.net.
example.net.      12h      IN      MX      10      mx2.example.net.
```

Über die Präferenz wird festgelegt, welcher der Mailserver bevorzugt genutzt werden soll (niedrigerer Wert = höhere Priorität). Der Wertebereich für die Präferenz ist 16 Bit groß, dennoch hat es sich durchgesetzt, als Präferenz immer Vielfache von 10 zu verwenden. Die Präferenz 10 hat also die höchste Priorität. Es ist durchaus möglich (und in der Regel auch sinnvoll), mehrere Server mit der gleichen Präferenz anzugeben. Der einliefernde Mailserver nimmt dann zufällig einen der Mailserver (dies wird als *DNS-Round-Robin* bezeichnet).

Ist dieser nicht erreichbar, wird ein anderer Mailserver der gleichen Priorität gewählt. Erst wenn alle MX-Server der gleichen Priorität durchprobiert wurden, darf der einliefernde Mailserver sich an Server einer niedrigeren Priorität wenden – dabei halten sich Spammer übrigens bewusst nicht an diese Regel. Viele Postmaster versäumen es nämlich leider, den sekundären Mailserver genauso gut zu sichern wie den primären. Daher versuchen Spammer oft gezielt, ihre Mails bei den MX-Servern mit der niedrigeren Priorität loszuwerden.

**Bei merkwürdigen Verbindungsversuchen RFC beachten!**

Beachten Sie, dass sich, wenn für eine Domain kein MX-Eintrag vorhanden ist, Mailserver laut RFC an den *A-Record* der Domäne wenden sollen! Falls Sie also ungewöhnliche Zugriffsversuche feststellen sollten, wird dies voraussichtlich am RFC-Standard liegen.

► CNAME-Record

Mit dem CNAME-Record (*Canonical Name Record*) können Aliase (auch als *Verweise* bezeichnet) definiert werden:

```
www.example.net.      3600    IN       CNAME    server01.example.net.
```

Dabei gibt es ein paar Besonderheiten, die Sie beachten müssen. Weder NS-Records noch Zonen selbst dürfen auf einen *CNAME* zeigen. Darüber hinaus sollte bei MX-Records auf *CNAME*-Records verzichtet werden.

23.1.5 Die erste vollständige Zone

Mit den bisher vorgestellten Resource Records lässt sich bereits eine komplette Zone definieren. In Listing 23.8 sehen Sie ein Beispiel für die Zone *example.net*:

```
$TTL 12h
@      IN      SOA    ns1.example.net. admin.example.net. (
                2023010701      ; Serial
                12h             ; Refresh
                1h              ; Retry
                10d             ; Expire
                1h              ; Negative Cache TTL
                )
      IN      NS     ns1.example.net.
      IN      NS     ns2.example.net.

      IN      MX     10      mx1
      IN      MX     10      mx2

ns1    IN      A      192.168.2.10
ns2    24h    IN      A      192.168.100.10
mx1    IN      A      192.168.2.11
mx2    IN      A      192.168.2.12
www    IN      CNAME  ns1
```

Listing 23.8 Beispiel für eine »forward«-Zone

Nach einem Neustart des Servers können Sie die Konfiguration mit dem *dig*-Kommando überprüfen:

```

root@satur:~# dig axfr example.net @localhost
; <<>> DiG 9.18.1-1ubuntu1.2-Ubuntu <<>> axfr example.net @localhost
; global options: +cmd
example.net.  43200  IN  SOA  ns1.example.net. admin.example.net. \
                2023010201 43200 3600 864000 3600
example.net.  43200  IN    NS   ns1.example.net.
example.net.  43200  IN    NS   ns2.example.net.
example.net.  43200  IN    MX   10 mx1.example.net.
example.net.  43200  IN    MX   10 mx2.example.net.
mx1.example.net. 43200  IN    A    192.168.2.11
mx2.example.net. 43200  IN    A    192.168.2.12
ns1.example.net. 43200  IN    A    192.168.2.10
ns2.example.net. 86400  IN    A    192.168.100.10
www.example.net. 43200  IN    CNAME ns1.example.net.
example.net.  43200  IN  SOA  ns1.example.net. admin.example.net. \
                2023010201 43200 3600 864000 3600

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(localhost) (TCP)
;; WHEN: Mon Jan 02 10:14:05 UTC 2023
;; XFR size: 11 records (messages 1, bytes 304)

```

Listing 23.9 Anzeige der »forward«-Zone mit »dig«

Die Option `axfr` des Befehls `dig` leitet einen Zonentransfer ein, daher wird die gesamte Zone geladen und dargestellt. Das ist für den Produktivbetrieb nicht unbedingt gewollt und muss noch eingeschränkt werden.



Mehr dazu finden Sie in Abschnitt 23.1.8, »Secondary-Server«. Zum Testen der ersten Konfiguration ist es aber ideal. Durch die Angabe von `@localhost` wird `dig` mitgeteilt, dass die Anfrage an `localhost` gesendet werden soll und nicht an den ersten Server aus der `/etc/resolv.conf`.

Wie Sie in Listing 23.9 sehen können, werden die optionalen Werte wie `IN` und `TTL` automatisch bei allen Einträgen ergänzt, und die Zeitangaben werden in Sekunden umgerechnet, da dies das Standardformat ist.

23.1.6 Die hint-Zone

Um einen voll funktionsfähigen Server aufzusetzen, der auch Anfragen rekursiv auflösen kann, die außerhalb der eigenen Zonen (*example.net*) sind, müssen wir dem Server noch mitteilen, wo er die Root-Server findet.

Diese Konfigurationen werden in der sogenannten *hint*-Zone vorgenommen. Analog zur Zone *example.net* muss diese in der *named.conf* angegeben werden:

```
zone "." {
    type hint;
    file "/usr/share/dns/root.hints";
};
```

Listing 23.10 Eintrag für die »hint«-Zone in der »named.conf«

Die dazugehörige Zonendatei befindet sich entweder unter `/usr/share/dns/root.hints`, unter `/var/named/named.ca` oder unter `/var/lib/named/root.hint` und wird mit dem BIND mitgeliefert. Aktuelle Versionen können von InterNIC unter www.internic.com/domain/named.root bezogen oder mittels `dig` abgefragt werden – setzen Sie dafür den Befehl `dig +bufsize=1200 +norec NS . @a.root-servers.net` ab (was zum Beispiel CentOS während der Installation durchführt). Listing 23.11 zeigt einen Ausschnitt aus der *hint*-Zone:

```
...
; FORMERLY NS.INTERNIC.NET
;
.                3600000      NS      A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET. 3600000      A       198.41.0.4
A.ROOT-SERVERS.NET. 3600000      AAAA    2001:503:ba3e::2:30
;
; FORMERLY NS1.ISI.EDU
;
.                3600000      NS      B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. 3600000      A       199.9.14.201
B.ROOT-SERVERS.NET. 3600000      AAAA    2001:500:200::b
[...]
```

Listing 23.11 Ausschnitt aus der »hint«-Zone

Nach Abschluss der Konfiguration können Sie probieren, externe Domains aufzulösen. In Listing 23.12 sehen Sie die Auflösung einer externen Domäne:

```
root@debian:~# dig www.rheinwerk-verlag.de @localhost

; <<<> DiG 9.16.33-Debian <<<> www.rheinwerk-verlag.de @localhost
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 46013
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 1232
;; COOKIE: 6e777cdb29499180100000063b2b1ed64f8977f12d2f271 (good)
;; QUESTION SECTION:
;www.rheinwerk-verlag.de. IN A
```

```
;; ANSWER SECTION:
www.rheinwerk-verlag.de. 86400 IN A 46.235.24.168

;; Query time: 0 msec
;; SERVER: ::1#53(::1)
;; WHEN: Mon Jan 02 11:29:01 CET 2023
;; MSG SIZE rcvd: 96
```

Listing 23.12 Auflösung einer externen Domäne

Herzlichen Glückwunsch, es funktioniert! Wie Sie in Listing 23.12 sehen können, hat *www.rheinwerk-verlag.de* die IP-Adresse 46.235.24.168. Die TTL des Eintrags beträgt 86.400 Sekunden, also 24 Stunden. Dass der Cache funktioniert, zeigt ein weiterer Aufruf:

```
root@saturn:~# dig www.rheinwerk-verlag.de @localhost
[...]
;; ANSWER SECTION:
www.rheinwerk-verlag.de. 86354 IN      A      46.235.24.168
[...]
;; Query time: 0 msec
[...]
```

Listing 23.13 Erneute Abfrage der externen Domäne

Die `Query time` der Anfrage war mit 0 Millisekunden recht schnell (üblicherweise kann die lokale Auflösung bis zu 5 Millisekunden in Anspruch nehmen), und der Timer der TTL steht bereits auf 86.354 Sekunden. Nach Ablauf dieser Zeit ist eine neue Anfrage bei einem der beiden für *www.rheinwerk-verlag.de* verantwortlichen Nameserver notwendig.

23.17 Reverse Lookup

Bisher haben wir lediglich Namen in IP-Adressen aufgelöst. Nun widmen wir uns der *Rückwärtsauflösung (Reverse Lookup)*, also der Auflösung der IP-Adresse in einen Domainnamen. Für einen DNS-Server sind alle Namen, auch IP-Adressen. Um diese effektiv auflösen zu können, wurde das Konzept des Reverse Lookup geschaffen. Dabei werden die Zonendateien, in denen die Zuordnung von IP-Adressen zu Namen vorgenommen wird, in umgekehrter Reihenfolge aufgebaut. Diese Zonen haben immer das Suffix *in-addr.arpa* – davon können Sie ableiten, wann diese Funktion implementiert wurde (richtig: zu Zeiten des ARPANET).

Ein Reverse Lookup von 194.8.219.19 wird über diesen Weg auf *19.219.8.194.in-addr.arpa* umgebogen. Auf diese Weise wird die sonst auch übliche baumartige DNS-Struktur aufgebaut. Daher können Teilnetze an unterschiedliche Nameserver delegiert werden. In vielen Fällen sorgen Nameserver Ihres Providers für die Rückwärtsauflösung, insbesondere dann, wenn Ihnen ein kleineres Netz als /24 zugeteilt wird. Es gibt zwar Möglichkeiten, auch kleinere Netze für Reverse Lookups zu delegieren, aber nicht alle Provider bieten das an.

In Listing 23.14 sehen Sie ein Beispiel für eine Rückwärtsauflösung:

```
root@saturn:~# host 194.8.219.19
19.219.8.194.in-addr.arpa domain name pointer mail.rheinwerk-verlag.de.
```

Listing 23.14 Reverse Lookup für 194.8.219.19 mit dem »host«-Kommando

Für die Auflösung Ihrer IP-Adressen in Namen müssen Sie zunächst eine Zone definieren und diese dann mit Informationen befüllen. Der dafür notwendige Resource Record heißt *PTR* (*pointer*, engl. für *Zeiger*) – siehe Listing 23.15:

```
zone "2.168.192.in-addr.arpa" {
    type primary;
    file "reverse/db.2.168.192";
};
```

Listing 23.15 Ergänzungen in der »named.conf«

Wie schon für die Forward-Zone müssen Sie auch für die Reverse-Zone eine Zonendatei erstellen. In unserem Beispiel ist das die Datei `/var/cache/bind/reverse/db.2.168.192`. Erstellen Sie diese Datei, und fügen Sie den Inhalt aus Listing 23.16 ein:

```
$TTL 12h
@      IN      SOA      ns1.example.net. admin.example.net. (
                                2          ; Serial
                                12h         ; Refresh
                                1h          ; Retry
                                10d         ; Expire
                                1h          ; Negative Cache TTL
                                )
                                IN      NS      ns1.example.net.
                                IN      NS      ns2.example.net.
10     IN      PTR      ns1.example.net.
11     IN      PTR      mx1.example.net.
12     IN      PTR      mx2.example.net.
```

Listing 23.16 Zonendatei für die »PTR-Records«

Mit dem `host`-Kommando können Sie nach einem Neustart des Nameservers die Konfiguration überprüfen. In Listing 23.17 sehen Sie eine Überprüfung der Rückwärtsauflösung:

```
root@saturn:~# host 192.168.2.10 localhost
Using domain server:
Name: localhost
Address: 127.0.0.1#53
Aliases:
10.2.168.192.in-addr.arpa domain name pointer ns1.example.net.
```

Listing 23.17 Beispiel für eine Rückwärtsauflösung

Damit auch das `host`-Kommando den lokalen Nameserver verwendet, müssen Sie diesen nach dem abzufragenden Namen angeben – also nicht wie bei `dig` mit einem führenden `@`-Zeichen versehen.

23.1.8 Secondary-Server

Hochverfügbarkeit muss bei Nameservern nicht über einen Cluster gelöst werden. Das Protokoll selbst liefert bereits alle Möglichkeiten, um die Verfügbarkeit gewährleisten zu können. Dabei ist Hochverfügbarkeit nicht nur in Ihrem eigenen Interesse zu gewährleisten, die *DE-NIC* schreibt für *de*-Domains zum Beispiel zwei bis fünf Nameserver vor. Dabei ist einer der Server stets der *Primary DNS* und damit der Taktgeber. Nur auf ihm werden die Zonendateien gepflegt. Die *Secondary DNS* oder Secondary-Server holen sich in regelmäßigen Abständen (siehe den SOA-Record) die Zonendateien vom Primary, antworten auf Anfragen für die Zone aber auch autoritativ.

Die Konfiguration eines Secondary-Servers ist denkbar einfach. Installieren Sie einen weiteren BIND-Server, und teilen Sie ihm in der *named.conf* mit, für welche Zonen er Secondary spielen soll. Bei der Zonenkonfiguration müssen Sie dann den Zonentyp auf `secondary` setzen. Zusätzlich muss mit `primaries` angegeben werden, wer der primäre Nameserver der Zone ist. In Listing 23.18 sehen Sie ein Beispiel für die Einträge in der *named.conf* eines Secondary-Servers:

```
zone "example.net" {
    type secondary;
    file "secondary/example.net";
    primaries { 192.168.2.10; };
};
```

Listing 23.18 Eintrag für eine Secondary-Zone in der »named.conf« auf dem Secondary-Server

Damit der Zonentransfer, also das Laden der Zone vom primären Nameserver, funktioniert, muss die bei `file` angegebene Datei für den BIND-Server beschreibbar sein. Auf dem Primary sind prinzipiell keine weiteren Konfigurationen notwendig. Um aber zu verhindern, dass jeder einfach Ihre Zonendateien abfragen kann, sollten Sie Zonentransfers unbedingt auf die Secondary-Server beschränken. Dafür müssen Sie im `options`-Block auf dem Primary mit der Direktive `allow-transfer` eine Liste von IP-Adressen setzen (siehe Listing 23.19).

```
options {
    directory "/var/cache/bind";
    listen-on-v6 { none; };
    allow-transfer { 127.0.0.1; 192.168.100.10; };
};
```

Listing 23.19 Einträge für den Zonentransfer auf dem Primary

Mit der Konfiguration aus Listing 23.19 erlauben Sie für alle Zonen den Transfer von 127.0.0.1 und 192.168.100.10. Falls Ihre Zonen von unterschiedlichen Nameservern gehalten werden, können Sie die Direktive auch in den zone-Blöcken setzen.

Nach einem Neustart des Nameservers wird der Zonentransfer automatisch angestoßen. Nach wenigen Sekunden haben Sie auf dem Secondary-Server dann im angegebenen Verzeichnis die Zone Ihres Primary-Servers. Sehen wir uns nun die Zonendatei auf dem Secondary-Server etwas genauer an (siehe Listing 23.20):

```
example.net.      43200 IN SOA ns1.example.net. admin.example.net.\
                               2023010203 43200 3600 864000 3600

example.net.      43200 IN NS ns1.example.net.
example.net.      43200 IN NS ns2.example.net.
example.net.      43200 IN MX 10 mx1.example.net.
example.net.      43200 IN MX 10 mx2.example.net.
mx1.example.net.  43200 IN A 192.168.2.11
mx2.example.net.  43200 IN A 192.168.2.12
ns1.example.net.  43200 IN A 192.168.178.137
ns2.example.net.  86400 IN A 192.168.178.146
www.example.net.  43200 IN CNAME ns1.example.net.
```

Listing 23.20 Ausgabe der »Secondary-Zone«

Wie Sie Listing 23.20 entnehmen können, sind die Zonendateien nicht zu 100 % identisch. Das ist darauf zurückzuführen, dass die Datei nicht einfach kopiert wird, sondern dass die Zone ausgelesen wird (wie mit dem dig-Kommando) und neu geschrieben wird. Trotz der anderen Darstellung ist das Ergebnis bei Abfragen aber identisch – wie so oft führen viele Wege nach Rom. Wie Ihnen sicherlich aufgefallen ist, wird die Zonendatei auf dem Secondary-Server nicht im Text-Format gespeichert, sondern im sogenannten »raw«-Format. Dieses ist leider nicht direkt lesbar und muss vorab konvertiert werden. Der Befehl aus Listing 23.21 wandelt die Zone *example.com* aus der Datei *example.com* um und speichert das Resultat unter dem Dateinamen *example.com.text*:

```
$ named-compilezone -f raw -F text -o example.net.text example.net example.net
```

Listing 23.21 Konvertierung der »Secondary-Zone« mittels »named-compilezone«

Bei Änderungen »Serial« anpassen!

Beachten Sie, dass der Secondary-Server nur Änderungen von Zonendateien lädt, bei denen sich die Angabe *Serial* verändert bzw. sich der Wert erhöht hat. Denn nur über diesen Wert stellt der Secondary-Server fest, ob es Änderungen in der Zone gibt. Wenn Sie *serial* nicht erhöhen, werden Ihre Änderungen auch nicht auf den Secondary übertragen.



23.1.9 DNS-Server und IPv6

Natürlich lässt sich der BIND auch als DNS-Server für IPv6 konfigurieren. In Listing 23.22 sehen Sie die entsprechenden Einträge der IPv6-Zonen in einer *named.conf*-Datei:

```
zone "example.net" in {
    type primary;
    file "primary/db.example";
};

zone "123.168.192.in-addr.arpa" in {
    type primary;
    file "primary/db.192.168.123";
};

zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.d.f.ip6.arpa" {
    type primary;
    file "primary/db.fd00";
};
```

Listing 23.22 »named.conf« für IPv6

Wie Sie in Listing 23.22 sehen, ist der Nameserver für die Forward-Zone *example.net*, die IPv4-Reverse-Zone für das Netz *192.168.123.0/24* und die IPv6-Reverse-Zone für das Netz *f:d::* zuständig. In Listing 23.23 sehen Sie die geänderte Forward-Zonendatei:

```
$TTL 1800      ; 30 minutes
example.net.  IN SOA  adminbuch.example.net. root.adminbuch.example.net. (
                                2021010502 ; Serial
                                600         ; Refresh (10 minutes)
                                200         ; Retry (3 minutes 20 seconds)
                                604800     ; Expire (1 week)
                                1800       ; negTTL (30 minutes)
                                )
              IN NS   ns1.example.net.
              IN NS   ns16.example.net.

ns1           IN     A       192.168.123.102
ns16          IN     AAAA    fd00::102
admin-repl    IN     A       192.168.123.103
admin-repl6   IN     AAAA    fd00::103
winclient     IN     A       192.168.123.121
winclient6    IN     AAAA    fd00::121
```

Listing 23.23 Forward-Zonendatei für IPv6

Für jeden Host gibt es hier jeweils einen IPv4- und einen IPv6-Eintrag. In Listing 23.24 sehen Sie den Aufbau der Reverse-Zonendatei:

```
$TTL 259200      ; 3 days
$ORIGIN 0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.d.f.ip6.arpa.
0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.d.f.ip6.arpa. IN SOA ns.example.net. root.example.net. (
    2023010201 ; Serial
    86400      ; Refresh (1 day)
    1800       ; Retry (30 minutes)
    172800    ; Expire (2 days)
    1800       ; negTTL (30 minutes)
)
                                IN      NS      ns16.example.net.

102                                IN      PTR      ns16.example.net.
103                                IN      PTR      admin-repl.example.net.
121                                IN      PTR      winclient6.example.net.
```

Listing 23.24 Reverse-Zonendatei für IPv6

Da in der Beispieldatei mit der Variablen `$ORIGIN` gearbeitet wird, ist es sehr einfach die Host-Einträge zu erstellen, da nur der eigentliche Hostteil der Adresse eingegeben werden muss. Wollen Sie aber die Adresse in der ausführlichen Form verwenden, hilft Ihnen das Programm `ipv6calc` mit dem Schalter `-r`, das für jede Distribution verfügbar ist:

```
root@ubuntu:~# ipv6calc -r fd00::103
no input type specified, try autodetection...found type: ipv6addr
3.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.d.f.ip6.int.
```

Listing 23.25 Berechnung der Reverse-Zonendatei mit »`ipv6calc`«

23.2 Vertrauen schaffen mit DNSSEC

Mit den *Domain Name System Security Extensions (DNSSEC)* wird das offene und nicht gerade vor Sicherheit strotzende DNS abgesichert. Dafür wird eine Vertrauenskette (beginnend bei der Root-Zone ».«) bis zum abgefragten Record erstellt. Darüber kann geprüft werden, ob eine Antwort auch wirklich vom zuständigen Nameserver beantwortet und auf dem Transportweg nicht verändert wurde. In diesem Abschnitt wollen wir Ihnen zeigen, wie DNSSEC arbeitet und wie Sie Ihre Zonen signieren, um sicher Auskunft geben zu können.

23.2.1 Die Theorie: »Wie arbeitet DNSSEC?«

Damit DNS-Abfragen überprüft werden können, wurden mehrere neue Record-Typen eingeführt. Über diese kann sichergestellt werden, dass eine Anfrage wirklich korrekt ist und vom

zuständigen autoritativen Nameserver stammt. Ähnlich wie beim HTTPS werden hierfür Signaturen verwendet (allerdings wird beim DNS die Kommunikation nicht verschlüsselt). Abbildung 23.3 zeigt, was im Hintergrund geschieht, um eine Abfrage zu überprüfen.

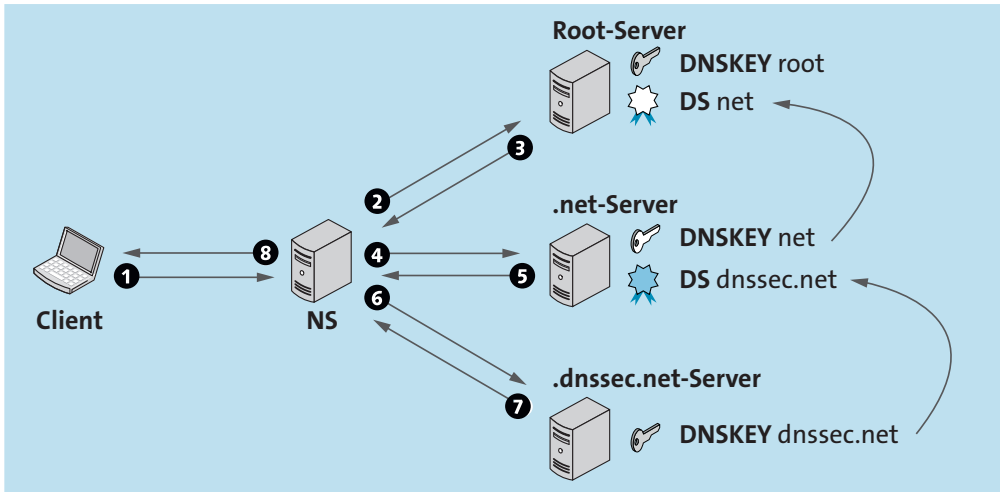


Abbildung 23.3 Ablauf der DNSSEC-Prüfung

Im ersten Schritt (1) fragt ein Client den lokalen Nameserver (NS) nach dem A-Record von `www.dnssec.net`. Der Nameserver beginnt nun mit der iterativen Namensauflösung. Dazu befragt er zunächst die Root-Zone (2). Da der Record mit DNSSEC abgesichert ist, liefert der Root-Server (3) bereits die dazugehörigen Records `DNSKEY` und `DS` mit. Nun fragt der Nameserver den für die Zone `.net` zuständigen Nameserver (4) und erhält erneut die Records `DNSKEY` und `DS` (5). Den Record `DNSKEY` kann der Nameserver nun mit dem vorherigen `DS`-Record abgleichen, um sicherzustellen, dass die Antwort valide ist. Dieser Prozess setzt sich nun fort (6 und 7), nämlich bei der Abfrage des autoritativen Nameservers von `dnssec.net`. Wenn alle Antworten valide waren, gibt der Nameserver die Antwort (8) an den Client weiter. Wie Sie sehen, ist das Vertrauensverhältnis hierarchisch; man spricht hier auch von einer *Chain of Trust*.

Neben den bereits gezeigten Records `DNSKEY` und `DS` gibt es noch weitere für DNSSEC notwendige Records. Diese schauen wir uns nun genauer an:

- ▶ **RRSIG (Resource Record Signature)**
enthält die kryptografische Signatur.
- ▶ **DNSKEY (DNS Public Key)**
enthält den öffentlichen Schlüssel.
- ▶ **DS (Delegation Signer)**
enthält den Hash zu einem `DNSKEY`.

- ▶ **NSEC und NSEC3 (Next Secure)**
wird zur eindeutigen Verneinung (engl. *denial-of-existence*) von einem Record verwendet – beim *NSEC3* kommen Hash-Werte statt Klartext zum Einsatz.
- ▶ **CDNSKEY und CDS**
wird zur Aktualisierung des *DS*-Records einer Kindzone (engl. *child zone*) in der Elternzone verwendet.

Die Signierung von Records findet gebündelt statt. Diese Bündelung nennt man *RRsets* (*Record Sets*). Dabei werden alle Records des gleichen Typs einer Zone zusammengefasst und signiert. Für die eigentliche Signierung kommen Schlüssel zum Einsatz (wie beim HTTPS). Beim DNS werden zwei Arten von Schlüsseln unterschieden:

- ▶ **ZSK (Zone-Signing Key)**
wird zur Signierung von *RRsets* (Zonen) verwendet.
- ▶ **KSK (Key-Signing Key)**
wird zur Signierung des ZSK verwendet.

Weshalb werden zwei Schlüssel verwendet? Dies hat den charmanten Vorteil, nicht unnötig viele Daten austauschen zu müssen. Der KSK ist eher statisch, da er nicht nur lokal, sondern auch beim übergeordneten Nameserver hinterlegt wird – ein Austausch ist relativ komplex. Der ZSK hingegen wird bei jeder Veränderung neu erstellt, der Austausch muss also zeitnah und einfach erfolgen. Daher wird der ZSK lokal verwendet und durch den KSK signiert. Durch die Vertrauensverkettung kann den ZSKs vertraut werden, wenn diese vom unveränderten KSK erzeugt wurden (was wiederum vom übergeordneten Nameserver abgefragt werden kann).

Dem aufmerksamen Leser wird nicht entgangen sein, dass die Vertrauenskette an einem Punkt endet: bei der Root-Zone. Weshalb sollten wir also der Root-Zone vertrauen? Hier kommt die *Root Signing Ceremony* ins Spiel. Bei dieser öffentlichen, überwachten und streng kontrollierten Zeremonie wird der Schlüssel erzeugt und in den Root-DNS-Servern eingespielt. Dieser Prozess wird alle paar Monate wiederholt und im Voraus geplant – hierzu finden Sie alle Daten unter <https://www.iana.org/dnssec/ceremonies>.

23.2.2 Anpassungen am Server

Damit Ihre Zone ebenfalls geprüft werden kann, müssen Sie ein paar Vorbereitungen treffen. In älteren Versionen des *bind* mussten Sie DNSSEC explizit aktivieren. Bei der Abfrage über den Server ist dies nicht mehr notwendig, da die Konfiguration bereits gesetzt ist (Konfiguration: `dnssec-validation auto;`).

Ob die Prüfung funktioniert, können Sie leicht mit der Testdomäne www.dnssec-failed.org überprüfen. In Listing 23.26 sehen Sie, wie sich das Ergebnis verändert, wenn die Option `dnssec-validation` auf dem Wert `auto` oder `off` steht.

```
### dnssec-validation auto
daniel@ubuntu:~$ dig @localhost www.dnssec-failed.org. A
[...]
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 26007
[...]
;www.dnssec-failed.org. IN A
[...]

### dnssec-validation off
daniel@ubuntu:~$ dig @localhost www.dnssec-failed.org. A
[...]
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26024
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
[...]
;; QUESTION SECTION:
;www.dnssec-failed.org. IN A

;; ANSWER SECTION:
www.dnssec-failed.org. 7200 IN A 68.87.109.242
www.dnssec-failed.org. 7200 IN A 69.252.193.191
[...]
```

Listing 23.26 Unterschiedliche Ergebnisse mit und ohne DNSSEC-Prüfung

Wie Sie sehen, erhalten Sie trotz fehlerhafter DNSSEC-Konfiguration eine »gültige« Antwort, wenn `dnssec-validation` auf den Wert `off` eingestellt ist.

23.2.3 Schlüssel erzeugen

Anschließend müssen Sie die benötigten Schlüssel erzeugen. Dafür verwenden wir das Programm `dnssec-keygen`. Zur Erzeugung eines ZSK führen Sie den Befehl aus Listing 23.27 aus:

```
daniel@dns:/var/cache/bind/primary$ sudo dnssec-keygen -a ECDSAP256SHA256 example.net
Generating key pair.
Kexample.net.+008+33733
```

Listing 23.27 Einen ZSK erzeugen mit »`dnssec-keygen`«

Mit den Parametern `-a ECDSAP256SHA256` wurde der Verschlüsselungsalgorithmus definiert. Standardmäßig wird ein 1024 Bit langer Schlüssel erstellt. Der Befehl erzeugt zwei Dateien, das sogenannte *key pair* (dt. *Schlüsselpaar*). Da das Programm im Verzeichnis `/var/cache/bind/primary` ausgeführt wurde, finden Sie dort nun die Dateien

Kexample.net.+008+33733.key (öffentlicher Schlüssel) und *Kexample.net.+008+33733.private* (privater Schlüssel). Die Benennung der Dateien folgt dieser Syntax:

```
K <ZONENAME> + <ALGORITHMUS> + <IDENTIFIER> . <TYP>
```

Listing 23.28 Syntax zur Benennung von Schlüsselpaaren bei DNSSEC

Alle Schlüssel fangen mit einem *K* an, gefolgt vom Zonennamen. Anschließend wird, durch ein Pluszeichen getrennt, der Algorithmus als vierstelliger Dezimalwert angegeben. Des Weiteren folgt, erneut durch ein Pluszeichen getrennt, eine Ziffer, die den Schlüssel identifiziert. Als Dateiendung wird stets der Typ verwendet, also *.key* für den öffentlichen Schlüssel und *.private* für den privaten Schlüssel. Anschließend können Sie mit dem gleichen Tool den KSK erzeugen, so wie in Listing 23.29 dargestellt:

```
daniel@dns:/var/cache/bind/primary$ sudo dnssec-keygen -a ECDSAP256SHA256 \
-f KSK example.net
```

```
Generating key pair.
Kexample.net.+008+57472
```

Listing 23.29 Einen KSK erzeugen mit »dnssec-keygen«

Die Befehle sind fast identisch: Für den KSK wurde lediglich der Parameter *-f KSK* hinzugefügt. Und da der KSK statisch ist, wurde er mit einer Länge von 2048 Bit erzeugt. Auch er erzeugt wieder den öffentlichen (*.key*) und den privaten (*.private*) Schlüssel.

23.2.4 Schlüssel der Zone hinzufügen und die Zone signieren

Die soeben erzeugten öffentlichen Schlüssel (*.key*) müssen Sie nun noch der bestehenden Zonendatei (*example.net*) hinzufügen. Dafür können Sie den Inhalt der Dateien einfach an die Zonendatei anhängen oder diese mit der Direktive *\$INCLUDE* inkludieren. In Listing 23.30 haben wir die zweite Methode eingesetzt:

```
$ echo "\$INCLUDE Kexample.net.+008+33733.key" >> example.net
$ echo "\$INCLUDE Kexample.net.+008+57472.key" >> example.net
```

Listing 23.30 Schlüssel an die Zone anhängen

Damit ist die Zone *example.net* zum Signieren vorbereitet. Das Signieren wird mit dem Programm *dnssec-signzone* vorgenommen. Dieses erwartet einige Parameter und Werte. Sehen wir uns daher zunächst die Syntax des Programms an:

```
$ dnssec-signzone -A -3 <SALT> -N INCREMENT -o <ORIGIN> -t <ZONE FILE>
```

Listing 23.31 Syntax von »dnssec-signzone«

Sehen wir uns nun die einzelnen Parameter und deren Werte etwas genauer an:

- ▶ -A
verhindert, dass bei der Erzeugung mit dem Verfahren *NSEC3* unsichere Delegationen ebenfalls signiert werden.
- ▶ -3 <SALT>
Mit diesem Parameter wird der beim Verfahren *NSEC3* benötigte *SALT* (engl. für *Salz*) als Hash-Wert angegeben. Dieser dient dazu, kryptografisch bessere Ergebnisse zu erzielen, da er bei der Erzeugung »eingestreut« wird.
- ▶ -N INCREMENT
Mit dieser Direktive weisen Sie *dnssec-signzone* an, die Angabe *Serial* der Zone beim Signieren direkt zu erhöhen, sodass Sie dies nicht nachträglich per Hand anpassen müssen.
- ▶ -o <ORIGIN>
gibt den Origin der Zone an – wenn keiner angegeben wird, wird der Dateiname als Ursprung verwendet (wie sonst beim DNS auch).
- ▶ -t
gibt am Ende des Verarbeitungsvorgangs eine Zusammenfassung aus.
- ▶ <ZONE FILE>
gibt die zu signierende Zonendatei an.

**Tipp: »SALT«**

Um einen guten *Salt* zu erzeugen, können Sie einfach für den Platzhalter <SALT> aus Listing 23.32 das nachstehende Kommando verwenden:

```
$(head -c 1000 /dev/random | sha1sum | cut -b 1-16)
```

Dieses liest 1000 Zeilen aus dem Zufallsgenerator *random*, berechnet den Hash-Wert und gibt eine 16-stellige Hash-Zahl aus.

In Listing 23.32 sehen Sie den zum bisherigen Beispiel passenden Befehl:

```
$ sudo dnssec-signzone -A -3 <SALT> -N INCREMENT -o example.net -t example.net
```

Verifying the zone using the following algorithms:

- ECDSAP256SHA256

Zone fully signed:

Algorithm: ECDSAP256SHA256: KSKs: 1 active, 0 stand-by, 0 revoked

ZSKs: 1 active, 0 stand-by, 0 revoked

example.net.signed

Signatures generated: 17

Signatures retained: 0

Signatures dropped: 0


```

Signatures successfully verified:      0
Signatures unsuccessfully verified:    0
Signing time in seconds:              0.015
Signatures per second:                1062.566
Runtime in seconds:                   0.043

```

Listing 23.32 Zone signieren mit »dnssec-signzone«

Der Befehl aus Listing 23.32 erzeugt die Datei *dsset-example.net*. (die im weiteren Verlauf relevant wird) und die Datei *example.net.signed*. Öffnen Sie diese im Editor Ihrer Wahl, um sich das bisherige Arbeitsergebnis ansehen zu können. Die signierte Zonendatei ist immens gewachsen. Hatte die Ausgangsdatei noch 24 Zeilen, so verfügt die signierte Zone über 132 Zeilen. Ohne die Hilfsprogramme ist ein Umgang mit DNSSEC kaum zu bewältigen.

23.2.5 Signierte Zone aktivieren

Damit Ihr DNS-Server die signierte Zone auch verwendet, müssen Sie ihm dies in der Datei *named.conf.zones* auch mitteilen. Passen Sie die Zonendefinition so an (siehe Listing 23.33):

```

zone "example.net" {
    type primary;
    file "/var/cache/bind/primary/example.net.signed";
};

```

Listing 23.33 Aktivieren der signierten Zone in »named.conf.zones«

Zu guter Letzt müssen Sie die Änderungen noch aktivieren. Starten Sie dafür den *bind* einmal neu. Ab jetzt ist Ihr DNS-Server in der Lage, für die Zone *example.net* mit DNSSEC zu antworten.

23.2.6 Signierung prüfen

Prüfen können Sie dies mit dem Alleskönner *dig*:

```

daniel@dns:~$ dig DNSKEY example.net. @localhost +multiline

; <<>> DiG 9.18.1-1ubuntu1.2-Ubuntu <<>> DNSKEY example.net. @localhost +multiline
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 59930
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 1232
;; COOKIE: 29f1c32edab29fbb0100000063b2dc4fec8c94037864959f (good)

```

```
;; QUESTION SECTION:
example.net. IN DNSKEY

;; ANSWER SECTION:
example.net. 43200 IN DNSKEY 256 3 13 (
/OBID1pR/ZSFpFuT5uniC8mQ7Q3pjIO/izEan8PH6fUG
jD9fss8iL/i3+nqPekhrD6oqgjWT3dJ0q5miG76MQg==
) ; ZSK; alg = ECDSAP256SHA256 ; key id = 46574
example.net. 43200 IN DNSKEY 257 3 13 (
uxNfn7iDBzqcBW0u1y+mA/s3gqEimlFmNekQuEheZ9/3
G/4ezNwW3lwvwSK62+TB0a57GDfuiCCoBjJOZ9BgHA==
) ; KSK; alg = ECDSAP256SHA256 ; key id = 29274

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(localhost) (UDP)
;; WHEN: Mon Jan 02 13:19:51 UTC 2023
;; MSG SIZE rcvd: 228
```

Listing 23.34 Prüfung von DNSSEC mit »dig«

Wie Sie in Listing 23.34 sehen, gibt der lokale DNS-Server den angefragten DNSSEC-Record korrekt aus. Dies ist aber nur die halbe Miete. Um zu prüfen, ob der DNS-Server wirklich DNSSEC spricht, müssen Sie das dig-Kommando so ausführen, wie in Listing 23.35 dargestellt:

```
daniel@dns:~$ dig A example.net. @localhost +noadditional +dnssec +multiline
; <<>> DiG 9.18.1-1ubuntu1.2-Ubuntu <<>> A example.net. @localhost +noadditional \
+dnssec +multiline
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56649
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
; COOKIE: 1ffe34d5118c1159010000063b2dc1924b0b38460c4da61 (good)
;; QUESTION SECTION:
example.net. IN A

;; AUTHORITY SECTION:
example.net. 3600 IN SOA ns1.example.net. admin.example.net. (
2023010208 ; serial
43200 ; refresh (12 hours)
3600 ; retry (1 hour)
864000 ; expire (1 week 3 days)
```

```

3600      ; minimum (1 hour)
)
example.net. 3600 IN RRSIG SOA 13 2 43200 (
20230201122731 20230102122731 46574 example.net.
Y5owL[...]i45fQ== )
VO39KG3CR3EQPOF3K37RLHEV308BIEMH.example.net. 3600 IN NSEC3 1 1 10 31600D06D564CC5D (
6I3RN[...]2E9234KB NS SOA MX RRSIG DNSKEY NSEC3PARAM )
VO39KG3CR3EQPOF3K37RLHEV308BIEMH.example.net. 3600 IN RRSIG NSEC3 13 3 3600 (
20230201122731 20230102122731 46574 example.net.
PgNT0[...]p1NrW== )

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(localhost) (UDP)
;; WHEN: Mon Jan 02 13:28:57 UTC 2023
;; MSG SIZE rcvd: 416

```

Listing 23.35 Prüfung der Signierung mit »dig«

23.2.7 Die Signierung veröffentlichen

Fertig? Noch nicht ganz. Damit auch die große weite Welt Ihren signierten Zonen vertraut, müssen Sie die *DS*-Records Ihrem Domänen-Registrar melden. Diese Records stehen in der ebenfalls von `dnssec-signzone` erzeugten Datei `dsset-example.net`. (siehe Listing 23.36):

```

daniel@saturn:/var/cache/bind/primary$ cat dsset-example.net.
example.net. IN DS 29274 13 2 F415[...]47DB E90CEEEC

```

Listing 23.36 Ausgabe der Datei »dsset-example.net.«

Wie Sie sehen, war `dnssec-signzone` so freundlich und hat den benötigten Record direkt in DNS-Schreibweise aufbereitet und abgelegt. Der Record-Typ *DS* verfügt über mehrere Werte, nämlich über den Identifier, den verwendeten Algorithmus, den *Digest*-Typ (jeweils als Dezimalwert) und den *Digest* selbst (ähnlich wie die Dateibenennung der Schlüsselpaare).

Nachdem Sie den *DS*-Record in Ihrem Domänen-Registrar eingespielt haben und dieser Ihre *DS*-Records aktiviert hat, haben Sie es geschafft. Mithilfe von `dig` können Sie dies selbstverständlich kontrollieren. Führen Sie dafür einfach das Kommando aus Listing 23.37 aus:

```

daniel@saturn:~$ dig +trace +noadditional DS example.net. @8.8.8.8 | grep DS

```

Listing 23.37 Prüfung der *DS*-Records mit »dig«

Der Befehl fragt den Google-DNS (8.8.8.8) nach dem Record-Typ *DS* und limitiert die Ausgabe auf den Inhalt der *DS*-Records. Beachten Sie dabei, dass Sie die Zone entsprechend anpassen.

23.2.8 Weniger anstrengend: Mehr Automatismus!

Natürlich geht's auch etwas bequemer und Sie müssen nicht bei jeder Konfigurationsanpassung die Zonen neu signieren. Dafür wurden im *bind* zwei Optionen geschaffen:

- ▶ `auto-dnssec maintain`
Mit dieser Option übernimmt der *bind* die DNSSEC-Signierung selbstständig. Wenn zum Beispiel eine Signatur ausläuft oder ein neuer Zonenschlüssel vorhanden ist, wird das Signieren ausgelöst.
- ▶ `inline-signing yes`
Diese Option weist *bind* zusätzlich an, bei einem Zonentransfer die geänderten Zonen-daten zu signieren. Allerdings werden damit nur neue unsignierte Einträge verarbeitet. Ältere Einträge werden, wenn sie ablaufen, durch das `auto-dnssec maintain` bei Bedarf neu signiert.

Diese Optionen müssen pro Zone definiert werden. Für unsere bisherige Beispielzone könnte die Konfiguration wie folgt aussehen:

```
zone "example.net" {
    type primary;
    file "primary/example.net";
    auto-dnssec maintain;
    inline-signing yes;
};
```

Listing 23.38 Zonen-Konfiguration mit Automatismen



Rechte beachten!

Bitte denken Sie daran, dass die Dateien für den Dienst lesbar sein müssen. Ansonsten kann der Automatismus leider nichts für Sie tun. Achten Sie daher stets darauf, dass die Schlüssel auch dem Benutzer gehören, unter dem der *bind* läuft.

Nach einem Neustart des Dienstes werden alle Signaturen für Sie angelegt. Die signierte Zonendatei können Sie leider nicht direkt öffnen, um das Ergebnis zu kontrollieren, aber selbstverständlich gibt es dafür ein Tool. Mithilfe von `named-checkzone` können Sie das Format in eine für Menschen lesbare Form umwandeln lassen, so wie in Listing 23.39 dargestellt:

```
$ named-checkzone -D -f raw example.net example.net.signed
```

Listing 23.39 Zonen-Konfiguration mit Automatismen

Dass der Automatismus wirklich funktioniert, können Sie an den Logmeldungen sehen. Nach einem Neustart des Dienstes sollten Sie im Log Meldungen finden wie die in Listing 23.40:

```
named[703]: zone example.net/IN (signed): sending notifies (serial 2023010213)
named[703]: zone example.net/IN (signed): reconfiguring zone keys
named[703]: zone example.net/IN (signed): next key event: 03-Jan-2023 13:03:06.544
```

Listing 23.40 Systemmeldung des »bind« mit aktiviertem Automatismus

23.2.9 Fazit

Das DNSSEC bläht die Zonendateien immens auf. Wenn Sie kein konkretes Ziel verfolgen, zum Beispiel die Absicherung Ihrer Mailserver mit DNSSEC, bringt Ihnen DNSSEC kaum Vorteile. Aufgrund seiner Komplexität empfehlen wir Ihnen dringend, über eine Managementlösung nachzudenken. Rein über die Konsole ist DNSSEC relativ zäh zu bewältigen, zumindest wenn mehr als eine Zone verwaltet werden soll.

23.3 Client-Anfragen absichern mit »DNS over HTTPS (DoH)«

Bisher haben wir lediglich mittels DNSSEC dafür gesorgt, dass Dritte nachvollziehen können, dass die Antwort auf eine Anfrage wirklich vom Eigentümer einer Domäne stammt. Allerdings können auch rekursive DNS-Abfragen mittlerweile verschlüsselt werden. Beim *DNS over HTTPS (DoH)* werden, wie der Name bereits verrät, die DNS-Anfragen von Clients durch HTTPS geleitet. Das übergeordnete Ziel dabei ist, die Privatsphäre und Sicherheit zu erhöhen. Wie Sie Ihrem *bind* das DoH beibringen, zeigen wir Ihnen in diesem Abschnitt.

23.3.1 Installation

Das DoH wird von *bind* ab der Version 9.17 unterstützt. Lediglich Ubuntu 22.04 stellt Ihnen bereits einen aktuelleren *bind* in den Standardpaketquellen zur Verfügung. Bei den übrigen Distributionen müssen Sie Hand anlegen, was wir Ihnen nun genau zeigen.

Debian

Aktivieren Sie zunächst die *backports*-Repositories in der Datei */etc/apt/sources.list*:

```
deb http://deb.debian.org/debian bullseye-backports main contrib non-free
deb-src http://deb.debian.org/debian bullseye-backports main contrib non-free
```

Listing 23.41 »Backports«-Repositories aktivieren in »/etc/apt/sources.list«

Anschließend müssen Sie die Paketquellen neu laden und können dann die Version 9.18 installieren (siehe Listing 23.42):

```
$ apt -t bullseye-backports upgrade
$ apt install -f bind9 bind9utils bind9-dnsutils -t bullseye-backports
```

Listing 23.42 Repositories aktualisieren und den »bind« installieren

Wichtig dabei ist die Angabe des Repositorys mit dem Schalter `-t`, da ansonsten die regulären Repositorys verwendet werden.

openSUSE Leap

Auch bei openSUSE müssen Sie ein alternatives Repository hinzufügen, die Paketquellen neu laden und können dann die neuere Version von *bind* installieren. Setzen Sie dafür die Kommandos aus Listing 23.43 ab:

```
$ zypper addrepo https://download.opensuse.org/repositories/network/15.4/network.repo
$ zypper refresh
$ zypper install bind-9.18.1-lp154.394.2.x86_64 bind-utils-9.18.1-lp154.394.2.x86_64
```

Listing 23.43 Repository ergänzen, aktualisieren und den »bind« installieren

Wenn Sie die Installation ohne die Versionsnummer ausführen, dann erhalten Sie einen Hinweis, dass eine neuere Version vorhanden ist – sogar inklusive des notwendigen Kommandos, um diese zu installieren.

CentOS

Für CentOS existiert leider kein Paket und auch auf den üblichen Plattformen (wie *ELRepo* oder *rpmfind.net*) wurde zur Drucklegung keine aktuellere Version als 9.16 angeboten. Hier müssen Sie den *bind* leider per Hand kompilieren, um das DoH nutzen zu können.

23.3.2 Vorbereitungen

Bevor wir uns um die eigentliche Konfiguration kümmern, müssen wir zunächst ein paar Vorbereitungen treffen. In diesem Fall wortwörtliche ein Paar, also zwei:

- ▶ Verzeichnisse erstellen: `/etc/<bind/named>/keys`
- ▶ Zertifikat erstellen

Den ersten Punkt können Sie schnell erledigen, den zweiten theoretisch auch. Da die gängigen Tools die Zertifikatskette nicht prüfen, sondern das HTTPS lediglich zur Verschlüsselung einsetzen, sind offizielle Zertifikate nicht zwingend erforderlich. Um aber potenzielle Fehler zu vermeiden und generell einen guten Stil zu wahren, sollten Sie dennoch welche einsetzen. Wir empfehlen Ihnen an dieser Stelle *Let's Encrypt*-Zertifikate – wie Sie solche beziehen können zeigen wir Ihnen ausführlich in Abschnitt 31.4, »Einmal mit allem und kostenlos bitte: Let's Encrypt«.

Natürlich können Sie für DoH auch Zertifikate Ihrer eigenen CA verwenden oder (wie wir im folgenden Beispiel) einfach selbst signierte verwenden – wir setzen dafür auf zwei »OpenSSL-One-Liner«, die sowohl die CA erzeugen als auch die benötigten Server-Zertifikate und -Schlüssel:

```
# CA erstellen
$ openssl req -x509 -sha512 -nodes -extensions v3_ca -newkey rsa:4096 \
-keyout MyRootCA.key.pem -days 7320 -out MyRootCA.cert.pem \
-subj "/C=DE/ST=NRW/L=Moers/O=Adminbuch-Ltd/CN=MyRootCA"

# Zertifikat erstellen
$ openssl req -new -newkey rsa:4096 -days 365 -nodes -x509 \
-subj "/C=DE/ST=NRW/L=Moers/O=Adminbuch-Ltd/CN=ns1.example.net" \
-reqexts SAN -extensions SAN -config <(cat /etc/ssl/openssl.cnf \
<(printf '[SAN]\nsubjectAltName=DNS:ns1.example.net')) \
-CA MyRootCA.cert.pem -CAkey MyRootCA.key.pem \
-keyout ns1.example.net.key.pem -out ns1.example.net.cert.pem
```

Listing 23.44 Selbst signierte CA und ein Zertifikat für DoH erstellen

Die beiden Kommandos erzeugen in dem Verzeichnis, in dem sie ausgeführt werden, die Dateien *MyRootCA.key.pem* (CA-Schlüssel), *MyRootCA.cert.pem* (CA-Zertifikat), *ns1.example.net.key.pem* (Serverschlüssel) und *ns1.example.net.cert.pem* (Serverzertifikat).

Bitte passen Sie bei dem Kommando aus Listing 23.44 unbedingt die Parameter *subj* an Ihre Gegebenheiten an, und passen Sie im zweiten Kommando auch das via *printf* inkludierte [*SAN*] an Ihren Servernamen an – selbstverständlich sollten Sie dann auch die Dateinamen der Parameter *keyout* und *keyout* anpassen. Kopieren Sie anschließend das Serverzertifikat und den Schlüssel nach */etc/bind/keys*.

Mehr zum Thema Zertifikate?

Alles zum Thema Zertifikate finden Sie in Kapitel 31, »Verschlüsselung und Zertifikate«.

23.3.3 Konfiguration

Die eigentliche Konfiguration besteht nur aus ein paar wenigen Zeilen, die Sie in der *named.conf* ergänzen müssen. Listing 23.45 zeigt die entsprechenden Zeilen:

```
// # Zertifikate & Schlüssel
tls local-tls {
    key-file "/etc/bind/keys/ns1.example.net.key.pem";
    cert-file "/etc/bind/keys/ns1.example.net.cert.pem";
};

// # HTTP-Endpunkt
http local-http-server {
    endpoints { "/dns-query"; };
};
```

```
options {
    [...]
    // # HTTPS-Ports
    https-port 443;

    // # IPv4 und IPv6
    listen-on port 443 tls local-tls http local-http-server { any; };
    listen-on-v6 port 443 tls local-tls http local-http-server { any; };
};
```

Listing 23.45 DoH in »named.conf« aktivieren

Bitte achten Sie darauf, dass der options-Block ergänzt werden muss. Nach dem obligatorischen Neustart des Dienstes können Sie Ihren DNS-Server mittels DoH abfragen.

23.3.4 Funktionstest

Um die Behauptung zu überprüfen, können Sie einfach das übliche Kommando dig verwenden. Mit dem Parameter +https stellt dig die DNS-Anfragen via HTTPS, so wie in Listing 23.46 dargestellt, an Ihren DNS-Server:

```
daniel@client:~# dig +https @ns1.example.net rheinwerk-verlag.de

; <<>> DiG 9.18.8-1~bpo11+1-Debian <<>> +https @ns1.example.net rheinwerk-verlag.de
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 18767
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 1a68e4ba0b45a04b0100000063b4271caad43559d88e5354 (good)
;; QUESTION SECTION:
;rheinwerk-verlag.de. IN A

;; ANSWER SECTION:
rheinwerk-verlag.de. 85428 IN A 46.235.24.168

;; Query time: 79 msec
;; SERVER: 192.168.178.137#443(ns1.example.net) (HTTPS)
;; WHEN: Mon Jan 02 14:30:04 UTC 2023
;; MSG SIZE rcvd: 96
```

Listing 23.46 DNS-Abfrage via HTTPS mit »dig«

**Mindestens Version 9.17 oder höher benötigt!**

Bitte beachten Sie, dass Sie mindestens die Version 9.17 oder höher von `dig` benötigen. Zur Drucklegung war diese lediglich in Ubuntu 22.04 nativ enthalten.

23.3.5 Client-Konfiguration

Um Linux-Clients in Ihrem Netzwerk beizubringen, Ihren DNS-Server via DoH abzufragen, genügt es, das Tool `dnscrypt-proxy` aus den Paketquellen zu installieren. Anschließend müssen Sie nur noch das zuvor erstellte CA-Zertifikat unter `/etc/ssl/certs/` ablegen und die Konfigurationsdatei `/etc/dnscrypt-proxy/dnscrypt-proxy.toml` anpassen. Diese muss mindestens die Zeilen aus Listing 23.47 enthalten, um Anfragen an den DNS-Server `ns1.example.net` zu stellen zu können:

```
# Empty listen_addresses to use systemd socket activation
listen_addresses = []
server_names = ['ns1.example.net']

#[sources]
# [sources.'public-resolvers']
# url = 'https://download.dnscrypt.info/resolvers-list/v2/public-resolvers.md'
# cache_file = '/var/cache/dnscrypt-proxy/public-resolvers.md'
# minisign_key = 'RWQf6LRCGA9i53mLYecO4IzT51TGPpvWucNSch1CBMOQTaLn73Y7GF03'
# refresh_delay = 72
# prefix = ''

[static]
  [static.'ns1.example.net']
    stamp = 'sdns://AgcAAAAAAAAADz[...]XVlcnk'
```

Listing 23.47 Konfiguration in »dnscrypt-proxy.toml«

Mit den Parameter `server_names` können der (oder die) lokalen Nameserver angegeben werden. Diese müssen wiederum über einen eigenen Eintrag unterhalb von `static`-Block verfügen. Wichtig ist, dass Sie den Block `sources` auskommentieren, da ansonsten eine Liste öffentlich verfügbarer Resolver aus dem Internet geladen und eingerichtet wird!

Den Parameter aus der letzten Zeile müssen Sie nun noch generieren. Dazu können Sie einfach das Online-Tool `https://dnscrypt.info/stamps/` verwenden – das Tool stammt im Übrigen aus der Feder der `dnscrypt-proxy`-Autoren. Dort müssen Sie lediglich unter `PROTOCOL` das `DNS-OVER-HTTPS (DOH)` auswählen und die Felder mit Ihren Daten ausfüllen – für unser Beispiel bis hierher sieht das so aus wie in Abbildung 23.4 dargestellt.

The screenshot shows a web browser window with the URL `dnscrypt.info/stamps/`. The page title is "Online DNS Stamp calculator". The interface includes a navigation menu with links for SOFTWARE, DNSCRYPT & DOH SERVERS, MAP, FAQ, PROTOCOL, and DNS STAMPS. The main content area contains a form with the following fields and options:

- Protocol:** DNS-over-HTTPS (DoH)
- IP Address (IPv6 addresses must be in [] brackets):** 192.168.2.10
- Stamp:** `sdns://AgcAAAAAAAAADE5Mi4xNjguMi4xMAA`
- Host name (vhost+SNI) and optional port number:** ns1.example.net
- Hashes (comma-separated):** (empty field)
- Path:** /dns-query
- Options (all checked):**
 - DNSSEC
 - No filter
 - No logs

Abbildung 23.4 Online-Generierung des »Stamp«

Nach einem Neustart des Diensts mittels `systemctl reload dnscrypt-proxy` stellt dieser nun unter der Localhost-Adresse `127.0.2.1` den DNS-Proxy bereit. Stellen Sie DNS-Anfragen an diesen werden diese via DoH an Ihren DNS-Server weitergeleitet. Dieses Tool können Sie auf jedem Client separat betreiben oder zentral bereitstellen – dafür genügt es, beim Parameter `listen_addresses` eine IP-Adresse anzugeben.

Kapitel 24

OpenSSH

In diesem Kapitel lernen Sie den fortgeschrittenen Umgang mit dem OpenSSH-Client sowie dem OpenSSH-Server.

Im Jahr 1969 wurde *telnet* (TELeNETwork) entwickelt. Es eröffnete die Möglichkeit, auf der Kommandozeile eines entfernten Rechners zu arbeiten, als säße man direkt davor. Sie können sich leicht vorstellen, dass *telnet* sich in Windeseile verbreitete und äußerst beliebt war, weil es das Ende der »Turnschuhadministration« einläutete. Sicherheitsbedenken spielten noch keine Rolle, da die Anzahl der vernetzten Rechner zu Anfang der 70er-Jahre des vergangenen Jahrhunderts noch ausgesprochen übersichtlich war. So störte es zunächst niemanden, dass die Anmeldung ebenso wie der Inhalt der *telnet*-Session unverschlüsselt übertragen wurde. Das änderte sich, als *telnet* in den 90ern immer öfter Opfer von Sniffer-Angriffen wurde. 1995 entwickelte der finnische Student Tatu Ylönen das SSH-Protokoll, und 1999 debütierte ein SSH-Fork (Code-Ableger) unter dem Namen *OpenSSH* als Bestandteil der OpenBSD-Distribution 2.6.

OpenSSH wurde auf eine Vielzahl anderer Betriebssysteme portiert. Wenn Sie sich auf einem Linux-System die Versionsnummer von OpenSSH anzeigen lassen, finden Sie darin ein »p«:

```
$ ssh -V
OpenSSH_8.4p1 Debian-5+deb11u1, OpenSSL 1.1.1n 15 Mar 2022
```

Listing 24.1 OpenSSH-Versionsnummer anzeigen lassen

An diesem Buchstaben erkennen Sie, dass es sich um eine portierte Version handelt. Auch die Versionsnummer von OpenSSL, dessen Bibliotheken die SSH-Tools benötigen, wird angezeigt.

24.1 Die SSH-Familie

Zur SSH-Familie gehören folgende Programme:

- ▶ *ssh*: ein Secure-Shell-Client als Ersatz für *telnet*, *rlogin* und *rsh*
- ▶ *scp*: Secure Copy, ersetzt *rcp*.
- ▶ *sftp*: Secure FTP

- ▶ `sshd`: der SSH-Daemon
- ▶ `ssh-keygen`: erzeugt Authentifizierungsschlüssel.
- ▶ `ssh-keyscan`: liest öffentliche Schlüssel ein.
- ▶ `ssh-agent`: speichert den privaten Schlüssel im Arbeitsspeicher.
- ▶ `ssh-add`: lädt weitere private Schlüssel in den `ssh-agent`.

24.1.1 Die Clients: `ssh`, `scp`, `sftp`

Diese Clientsoftware ist Ihr Standardwerkzeug im Administrationsalltag. Schauen wir uns die Programme der Reihe nach an!

`ssh`

Im einfachsten Fall können Sie sich mit dem folgenden Kommando auf ein entferntes System einloggen:

```
$ ssh server.example.com
```

Listing 24.2 Login

Anstelle des Namens können Sie selbstverständlich auch die IP-Adresse eingeben. Wenn Sie sich zum ersten Mal mit diesem Server verbinden, kennt Ihr System natürlich den öffentlichen Schlüssel des Servers noch nicht, und Sie bekommen zunächst eine Folge von Hexadezimalzahlen präsentiert, etwa so:

```
Authenticity of host 'server.example.com (10.10.47.11)' can't be established.  
RSA key fingerprint is 27:53:1c:21:8a:12:db:aa:c2:2a:ec:55:a1:b6:06:97.  
Are you sure you want to continue connecting (yes/no)?
```

Listing 24.3 Fingerprint des öffentlichen Serverschlüssels

Das ist der »Fingerabdruck« (*Fingerprint*) des öffentlichen Serverschlüssels. Sie müssen den Schlüssel durch Eingabe von »yes« akzeptieren, um fortfahren zu können. Der Server wird Sie nun nach Ihrem Passwort fragen, und wenn Sie es korrekt eingeben, erhalten Sie eine Shell auf dem Remote-System. Das funktioniert aber nur, sofern Sie auf dem lokalen und dem entfernten System den gleichen Benutzernamen haben. Falls nicht, müssen Sie dem Remote-System mitteilen, mit welchem Benutzernamen Sie sich einloggen möchten:

```
# ssh peer@server.example.com
```

Listing 24.4 Login mit Benutzernamen

Den öffentlichen Schlüssel des Servers legt Ihr System in der Datei `~/.ssh/known_hosts` ab. Der öffentliche Schlüssel des Servers wird dabei von Ihrem System in der Datei `~/.ssh/known_hosts` abgelegt.

**Wenn sich der Schlüssel ändert**

Bei allen weiteren Verbindungsversuchen wird Ihr System prüfen, ob sich der Schlüssel geändert hat, und in diesem Fall wird es eine eindringliche Warnung ausgeben. Dass sich der Schlüssel geändert hat, kann möglicherweise darauf hindeuten, dass jemand versucht, Ihre SSH-Verbindung zu kapern oder umzuleiten. Es kann jedoch auch ganz harmlose Ursachen haben. Der Server ist möglicherweise neu installiert worden, oder die IP-Adresse hat sich geändert. In jedem Fall ist es legitim und ratsam, sich beim Administrator des Servers zu versichern, dass die Gründe für den Schlüsselwechsel harmlos sind.

24

Parameter wie Ihren Usernamen oder den Port, falls dieser vom SSH-Standard-Port 22 abweicht, können Sie in der Datei `~/.ssh/config` ablegen, um sie nicht bei jedem Verbindungsaufbau neu eintippen zu müssen. Diese Datei kann wie folgt aussehen:

```
Host example.com
    HostName 10.10.47.11
    Port 2222
    User peer
Host example.net
    HostName 10.10.47.42
    Port 443
    User backup
```

Listing 24.5 SSH-Client-Konfigurationsdatei

Mit dieser Konfigurationsdatei tippen Sie nur noch:

```
# ssh example.com
```

Listing 24.6 SSH-Login, Kurzfassung

Wollen Sie auf dem entfernten System nur einen einzigen Befehl ausführen, können Sie SSH diesen Befehl auch direkt mit auf den Weg geben:

```
# ssh example.com uptime
13:45:56 18 Tage 2:48 an, 1 Benutzer, Durchschnittslast: 0,09, 0,03, 0,00
```

Listing 24.7 SSH-Login mit Befehlsübergabe**»scp«**

Secure Copy (`scp`) funktioniert grundsätzlich wie der bekannte `cp`-Befehl, allerdings über Rechnergrenzen hinweg.

Wollen Sie die Datei `test.txt` aus dem aktuellen Verzeichnis auf Ihren Server `doku.example.com` und dort in das Verzeichnis `/usr/local/texte` kopieren, lautet das Kommando:

```
# scp test.txt doku.example.com:/usr/local/texte/
```

Listing 24.8 Eine Datei mit »scp« kopieren

Falls Sie auf dem Server einen anderen Benutzernamen verwenden als auf Ihrem lokalen System, müssen Sie diesen zusätzlich angeben:

```
# scp test.txt username@doku.example.com:/usr/local/texte/
```

Listing 24.9 Eine Datei als anderer User mit »scp« kopieren

scp unterstützt auch das rekursive Kopieren ganzer Verzeichnisbäume. Geben Sie dazu den Parameter `-r` (rekursiv), eventuell ergänzt um `-p` (preserve) an, wenn Sie auch Dateirechte etc. erhalten möchten:

```
# scp -rp /home/user/meinetexte/* username@doku.example.com:/usr/local/meinetexte/
```

Listing 24.10 Einen Verzeichnisbaum kopieren

sftp

Auch sftp benutzt die gleiche Syntax wie sein herkömmliches Pendant, ftp. Hier ist ein Beispiel für den Beginn einer SFTP-Sitzung:

```
peer@client:~$ sftp peer@storage.example.com
Connecting to storage.example.com...
peer@storage.example.com's password: *****
```

```
sftp> help
Available commands:
bye                    Quit sftp
cd path                Change remote directory to 'path'
chgrp [-h] grp path   Change group of file 'path' to 'grp'
[...]
```

Listing 24.11 Eine SFTP-Sitzung

24.1.2 Der Server: sshd

Die Konfigurationsdatei, mit der Sie das Verhalten des SSH-Daemons steuern, heißt `/etc/ssh/sshd_config`. Die in dieser Datei hinterlegten Default-Einstellungen sind vernünftig gewählt, und für die meisten Anwendungsfälle müssen Sie hier keine Änderungen vornehmen. Werfen wir trotzdem einen Blick auf die wichtigsten Schraubchen, an denen Sie im Bedarfsfall drehen können.

Benutzer und Gruppen

Sie können den `sshd` anweisen, nur bestimmte Benutzer zuzulassen oder solche, die sich in bestimmten Gruppen befinden.

Mit den folgenden Zeilen gestatten Sie nur den Benutzern »meier«, »mueller« und Mitgliedern der Gruppe »backup« den Login:

```
AllowUsers meier mueller
AllowGroups backup
```

Listing 24.12 Einzelne Nutzer und Gruppen zulassen, alle anderen verbieten

Umgekehrt geht's auch: Sie können mit `DenyUsers` und `DenyGroups` pauschal alle Nutzer und Gruppen zulassen, mit Ausnahme derer, die Sie explizit benennen:

```
DenyUsers schmitz schulze
DenyGroups webdesigner marketing1
```

Listing 24.13 Alle Nutzer und Gruppen zulassen, aber einzelne verbieten

Die Benutzer- und Gruppendifferenzen werden in folgender Reihenfolge geprüft:

1. `DenyUsers`
2. `AllowUsers`
3. `DenyGroups`
4. `AllowGroups`

Keine root-Logins

In der Regel ist es nicht notwendig, dass man sich mit root-Rechten auf einem System einloggen muss. Ein User-Login reicht aus. Von dort können Sie immer noch in den root-Account wechseln. In der `/etc/ssh/sshd_config` verbieten Sie root-Logins mit dieser Zeile komplett:

```
PermitRootLogin no
```

Listing 24.14 Keine root-Logins zulassen

Wenn Sie – wie gleich auf den nächsten Seiten gezeigt wird – einen SSH-Schlüssel für root erzeugt und auf dem Server hinterlegt haben, können Sie auch überlegen, den root-Login (nur) mit SSH-Schlüssel zuzulassen, d. h., den passwortbasierten root-Login zu verbieten:

```
PermitRootLogin prohibit-password
```

Listing 24.15 Keine root-Logins mit Passwort zulassen

Soll sichergestellt sein, dass alle Nutzer des Systems ausschließlich über SSH-Schlüssel Zugriff bekommen, können Sie auf dem SSH-Server den Passwort-Login ganz deaktivieren:

```
PasswordAuthentication no
```

Listing 24.16 SSH-Logins mit Passwort sind gar nicht mehr möglich.

¹ Kleine Gehässigkeiten erhalten die Feindschaft.

Protokollversionen

Ein aktueller *sshd* unterstützt in seiner Konfiguration serienmäßig nur noch die Protokollversion SSHv2, was Sie an der folgenden Zeile in der Konfigurationsdatei */etc/ssh/sshd_config* erkennen:

```
Protocol 2
```

Listing 24.17 Die Protokollversion 2 ist Standard.

Wenn Sie noch ein archäologisch interessantes Linux mit einer sehr alten SSH-Implementa-tion betreiben, das aber auf Ihren Server zugreifen können muss, können Sie dem SSH-Server die Erlaubnis geben, zusätzlich SSHv1 anzubieten. Dazu ändern Sie die Protocol-Zeile so:

```
Protocol 2,1
```

Listing 24.18 Die Protokollversion 1 wird zusätzlich erlaubt.

Der SSH-Server wird bei einer Clientanfrage zunächst das SSHv2-Protokoll anbieten und nur auf SSHv1 zurückfallen, falls die Anfrage misslingt.

24.2 Schlüssel statt Passwort

Login-Passwörter sind ein notorischer Schwachpunkt in jedem Sicherheitskonzept, weil sie oft zu einfach gewählt sind und durch Wörterbuchattacks gebrochen werden können. Mit OpenSSH können Sie sich auf einen entfernten Server einloggen, ohne ein Passwort eingeben zu müssen – Ihr SSH-Key genügt. Und bequemer ist es auch noch.

24.2.1 Schlüssel erzeugen

Achten Sie darauf, einen kryptografisch starken Schlüssel zu benutzen. Früher wurden per Default DSA-Schlüssel erzeugt, was als nicht mehr ausreichend sicher gilt und von heutigen OpenSSH-Versionen ohne Konfigurationsanpassungen auch nicht mehr als Login-Mög-lichkeit zugelassen wird. Heutige SSH-Versionen verwenden stattdessen als Standard RSA-Schlüssel. Um die Sicherheit zu erhöhen, sollten Sie jedoch explizit Keys mit mit 4096 Bit Länge erzeugen:

```
ssh-keygen -b 4096
```

Listing 24.19 Ein Schlüsselpaar mit RSA erzeugen

Noch besser als RSA-Schlüssel sind heutzutage jedoch Schlüssel mit dem Verfahren *Curve25519*, auch *ED25519* genannt: Diese basieren auf elliptischen Kurven und sind gleich-zeitig besonders sicher, klein und schnell. Insofern darf man sich bei diesen Krypto-Ver-fahren auch nicht von der Länge des Schlüssels irritieren lassen. Diese ist mit klassischen

DSA- oder RSA-Verfahren nicht zu vergleichen. Sehr alte Linux-Systeme werden ED25519 noch nicht unterstützen, alle modernen Distributionen haben damit jedoch kein Problem mehr. Statt des RSA-Schlüssels könnten Sie sich also eventuell gleich (auch oder nur) einen ED25519-Key erzeugen lassen – und im Notfall können Sie auch beides parallel verwenden:

```
ssh-keygen -t ed25519
```

Listing 24.20 Ein Schlüsselpaar mit ED25519 erzeugen

24

Das Programm fragt Sie nun, wohin es Ihre Schlüssel schreiben soll. Der Default-Pfad ist `~/.ssh/id_rsa` für den privaten Schlüssel und `~/.ssh/id_rsa.pub` für den öffentlichen Schlüssel – beziehungsweise `~/.ssh/id_ed25519` und `~/.ssh/id_ed25519.pub` bei ED25519-Schlüsseln. In der Regel werden Sie keine Veranlassung haben, diese Pfade zu ändern, und können sie einfach bestätigen. Danach werden Sie nach einer Passphrase gefragt. Diese Passphrase schützt den Zugriff auf Ihren Schlüssel und sollte nicht zu einfach gestrickt sein.

Wenn Sie ein gutes Passwort suchen, aber keinen Zufallsgenerator benutzen möchten, probieren Sie folgende Vorgehensweise aus: Denken Sie sich einen (längeren) Satz aus, den Sie sich gut merken können, etwa: »Montags lerne ich Latein, dienstags Geografie und mittwochs Mathematik, aber am Donnerstag habe ich frei!« Nun nehmen Sie von diesem Satz die Anfangsbuchstaben jedes Worts und die Satzzeichen. Sie erhalten: »MliI,dGumM,aaDhif!«. Das ist noch kein perfektes, aber ein brauchbares Passwort. Haben Sie die Passphrase eingegeben, wird Ihr Schlüsselpaar generiert und im gewählten Pfad abgelegt.



24.2.2 Passwortloses Login


Nun müssen Sie Ihren öffentlichen Teil des Schlüssels auf dem Server deponieren, auf dem sie sich per SSH anmelden möchten. Dazu verwenden Sie das Kommando `ssh-copy-id`:

```
# normalerweise:
ssh-copy-id username@remotehost

# nur falls der Public Key nicht im Standard-Pfad liegt:
ssh-copy-id -i /pfad/zum/public.key username@remotehost
```

Listing 24.21 SSH-Copy-ID

Ihr öffentlicher Schlüssel wurde nun der Datei `/home/username/.ssh/authorized_keys` auf dem Server `remotehost` hinzugefügt. Steht Ihnen `ssh-copy-id` einmal nicht zur Verfügung oder möchten Sie einmal aus einer Mail oder dem Clipboard heraus einen Schlüssel hinzufügen, können Sie ihn auf dem Server manuell ergänzen. Jeder Schlüssel wird dabei als Einzeiler hinzugefügt. Wenn Sie sich nun noch einmal auf dem Server einloggen, werden Sie nicht mehr nach Ihrem Login-Passwort gefragt. Sie werden aber nach der Passphrase für Ihren Schlüssel gefragt. Auch diese Passwortabfrage können Sie noch eliminieren, indem

Sie – was ausdrücklich nicht empfohlen wird – mit `ssh-keygen` ein Schlüsselpaar ohne Passphrase erzeugen. Dazu drücken Sie einfach , wenn `ssh-keygen` Sie nach der gewünschten Passphrase fragt.



Diese Vorgehensweise ist gefährlich! Wer es schafft, Ihren passwortlosen Schlüssel zu stehlen, kann damit problemlos Ihre Identität vortäuschen. Sie befinden sich nun in einem Habitat, das jedem Sysadmin vertraut ist, nämlich im Spannungsfeld zwischen Sicherheit und Bequemlichkeit. Aber entspannen Sie sich: Es gibt eine Lösung, und zwar in Gestalt des SSH-Agenten, den Sie im nächsten Abschnitt kennenlernen.

Zugriffsbeschränkungen in der `authorized_keys`

Nutzer, die nur wenige genau definierte Kommandos auf einem entfernten System ausführen müssen, benötigen dafür nicht unbedingt einen vollständigen Shell-Zugang. Durch eine Ergänzung in der `.ssh/authorized_keys` lässt sich der verfügbare Befehlssatz einschränken.



Nehmen wir als Beispiel an, ein Benutzer soll lediglich den Befehl `ls` ausführen dürfen. In diesem Fall stellen Sie dem Schlüssel des Benutzers einfach `command=/bin/ls` voran:

```
command="/bin/ls" ssh-rsa AAAAB3NzaC1yc2EAAA[...]  
5GzqcTdUwqxJ rob@funghi
```

Listing 24.22 Der Benutzer »rob« darf nur das Kommando »ls« ausführen.

Rob ruft nun auf seinem Client das Kommando `ls` wie folgt auf:

```
ssh root@server "ls"
```

Listing 24.23 Rob benutzt das einzige ihm erlaubte Kommando.

24.2.3 Der SSH-Agent merkt sich Passphrasen

Der SSH-Agent befreit Sie von der lästigen Pflicht, jedes Mal die Passphrase eingeben zu müssen, wenn Sie Ihren Schlüssel benutzen. So starten Sie den Agenten:

```
# Start:  
ssh-agent
```

```
# Start mit Passphrase-Timeout (nach Ablauf der angegebenen Zeit "vergisst"  
# der Agent Ihre Passphrase:  
ssh-agent -t 60m
```

```
# Agent stoppen:  
ssh-agent -k
```

Listing 24.24 Den SSH-Agenten starten und stoppen

Viele Distributionen starten den SSH-Agenten automatisch. Ob der Agent bereits läuft, finden Sie entweder durch einen Blick in die Prozessliste heraus oder durch diesen Befehl:

```
echo $$SSH_AGENT_PID
```

Listing 24.25 Lauft der SSH-Agent?

Gibt das Kommando eine Prozess-ID zuruck, lauft der SSH-Agent. Mit dem Kommando `ssh-add` konnen Sie eine oder mehrere Schlusselpassphrasen zum Gedachtnis des Agenten hinzufugen, wieder entfernen und sich die bereits »gelernten« Phrasen anzeigen lassen:

```
# Der SSH-Agent soll sich die Passphrase Ihres Standard-Schlussels merken:
ssh-add
```

```
# Der Agent soll die Passphrase wieder vergessen:
ssh-add -d
```

```
# Fingerprints aller Schlussel anzeigen lassen, die der Agent zu diesem
# Zeitpunkt kennt:
ssh-add -l
```

```
# Der Schlussel liegt nicht im ublichen Pfad (~/.ssh/):
ssh-add [-d] /pfad/zum/schluesse1
```

Listing 24.26 Funktionen von »ssh-add«

24.3 X11-Forwarding

Das X11-Forwarding ermoglicht es Ihnen, auf einem entfernten Rechner per SSH ein grafisches Programm zu starten, das auf Ihrem lokalen Display eingeblendet wird. Kontrollieren Sie zunachst, ob auf dem Server (also dem entfernten Rechner, auf dem die Anwendung tatsachlich lauft) in der Konfigurationsdatei `/etc/ssh/sshd_config` der Eintrag `X11Forwarding` existiert und auf `yes` gesetzt ist:

```
X11Forwarding yes
```

Listing 24.27 »/etc/ssh/sshd_config«: X11-Forwarding serverseitig erlauben

Nach der anderung ist ein Neustart des SSH-Daemons fallig. Stellen Sie nun sicher, dass auf dem Server `xauth` installiert ist. Danach kommt der Client an die Reihe: Auch diesem mussen Sie erlauben, X11-Forwarding zu verwenden. Auch dem Client sind zwei Eintrage in der Datei `/etc/ssh/ssh_config` in der Host-Sektion erforderlich:

```
Host *
    ForwardX11 yes
    ForwardX11Trusted yes
```

Listing 24.28 »/etc/ssh/ssh_config«: X11-Forwarding clientseitig vorbereiten

Nach diesen Vorbereitungen können Sie jetzt ein grafisches Programm eines Remote-Rechners auf Ihrem eigenen Desktop nutzen:

```
ssh -X user@remote.example.com firefox
```

Listing 24.29 Grafische Anwendung mit X11-Forwarding starten



Portforwarding kann ein Sicherheitsrisiko sein

So nützlich ein SSH-Portforwarding manchmal auch sein kann – Sicherheitsexperten sehen diese Funktion kritisch, denn der entfernte Server kann dann unter anderem Tastatureingaben lesen und so ggf. auch in den Besitz eingetippter Passwörter gelangen. Aktivieren Sie diese Funktion darum nur, wenn Sie sie auch tatsächlich benötigen, und schalten Sie sie schon auf Ihrem Client ab, wenn klar ist, dass nicht.

24.4 Portweiterleitung und Tunneling

Stellen Sie sich folgende Situation vor: Sie befinden sich im Urlaub und nutzen mit Ihrem Laptop den Internetzugang, den das Hotel Ihnen anbietet. Nun erfordert es die Situation, dass Sie auf einen MySQL-Server in Ihrem Firmennetz zugreifen. Leider ist der Zugriff dort hin nur aus dem Firmennetz erlaubt, Verbindungen aus dem Internet werden verworfen. Wenn es einen Rechner im Firmennetz gibt, den Sie per SSH erreichen können, ist das Problem lösbar. Sie können diesen Rechner anweisen, als Mittelsmann zwischen Ihrem Laptop und dem MySQL-Server zu agieren. Drei Rechner sind also beteiligt:

- ▶ Ihr Laptop (*laptop.hotel.local*) im Hotelnetz mit Internetzugang.
- ▶ Der MySQL-Server (*mysql.example.com*) im Firmennetz – dieser ist aus dem Internet nicht erreichbar.
- ▶ Ein Gateway-Server (*gate.example.com*) befindet sich im Firmennetz und ist aus dem Internet per SSH erreichbar.

Das folgende Kommando ermöglicht den Zugriff auf den MySQL-Server:

```
ssh -L 3306:mysql.example.com:3306 username@gateway.example.com
```

Listing 24.30 Portweiterleitung durch einen SSH-Tunnel

Die Syntax ist nicht gerade selbsterklärend, deshalb schauen wir uns das Kommando noch einmal Stück für Stück an:

- ▶ `ssh -L`: Der Parameter `-L` bestimmt die »Richtung« des Tunnels, in diesem Fall vom lokalen Rechner zum entfernten Server. Mit `-R` würden Sie einen Tunnel in der entgegengesetzten Richtung öffnen, sodass der Server dann einen Dienst auf Ihrem lokalen Rechner nutzen könnte.

- ▶ 3306: Dies ist der Port, der auf Ihrem lokalen Rechner geöffnet wird, also der »Eingang« des Tunnels.
- ▶ `mysql.example.com`: Dies ist der Server, auf dem der Dienst läuft, den Sie über den Tunnel ansprechen wollen.
- ▶ `:3306`: Dies ist der Port, auf dem der Dienst läuft, den Sie nutzen möchten. 3306 ist der Standardport für MySQL.
- ▶ `username@gateway.example.com`: Dies ist Ihr Login-Name und der Name des Servers, den Sie vom Hotel aus per SSH erreichen und damit als Vermittler nutzen können.

Wenn Sie nun den Befehl `mysql -h localhost -u username -p` eingeben, landen Sie auf `mysql.example.com`, der Sie nach Ihrem MySQL-Passwort fragt. Die Verbindung von Ihrem Laptop zum Gateway-Server ist SSH-typisch verschlüsselt, die Verbindung vom Gateway zum MySQL-Server allerdings nicht mehr.

Schauen Sie sich OpenVPN an

So praktisch ein Port-Forwarding mit SSH auch ist: In fast allen vorstellbaren Fällen ist ein VPN-Tunnel die bessere Lösung. Abschnitt 30.8, »OpenVPN«, zeigt Ihnen, wie Sie schnell und sicher Ihr eigenes VPN bauen.



24.4.1 SshFS: Entfernte Verzeichnisse lokal einbinden

SshFS (SSH Filesystem) ermöglicht es, ganze Verzeichnisbäume eines entfernten Servers lokal einzubinden. Es ist ein FUSE-Dateisystem (*Filesystem in Userspace*) und gehört zum Standardumfang aller großen Distributionen. Die Syntax ist recht einfach:

```
sshfs user@server:pfad mountpoint
```

Listing 24.31 Grundsätzliche Syntax des SshFS-Kommandos

Ein konkretes Beispiel: Der Benutzer Rob möchte das Verzeichnis `/opt/scripts` auf dem Server `work.example.com` lokal unter `/home/rob/work` einbinden. Das funktioniert mit dem folgenden Kommando:

```
sshfs rob@work.example.com:/opt/scripts /home/rob/work
```

Listing 24.32 Ein entferntes Verzeichnis lokal einbinden

Rob kann nun in diesem Verzeichnis so arbeiten, als lägen die Dateien lokal vor. Nach getaner Arbeit kann Rob das Verzeichnis mit dem Kommando `fusermount -u /home/rob/work` wieder aushängen. Wie flüssig das vonstattengeht, hängt natürlich von der Geschwindigkeit der Netzanbindung ab. Aber klar ist auch: Ein SshFS ist einfach und gut, aber ein gewisser Workaround. Echte spezialisierte Netzwerk-Dateisysteme sind performanter.

Kapitel 25

Administrationstools

Manchmal gibt es unscheinbare Programme, die bei näherer Betrachtung sehr nützlich sind. Dieses Kapitel nimmt sich deshalb bekannter, aber leider selten im Detail betrachteter Tools an, die Ihnen Ihren Arbeitsalltag vereinfachen können.

Viele Administratoren verwenden zur Problemlösung meist nur einen Weg – und zwar den, der ihnen bekannt ist. Viele Tools auf Linux-Systemen haben aber mehr drauf, als viele wissen. Ein paar dieser Tools haben wir Ihnen im Verlauf dieses Buches bereits vorgestellt. In diesem Kapitel wollen wir Ihnen die gängigsten Tools zeigen, die viele von Ihnen vermutlich bereits kennen, aber nicht in der vorgestellten Art oder in der vorgestellten Funktionsweise.

25.1 Was kann dies und jenes noch?

In diesem Abschnitt stellen wir Ihnen die üblichen Verdächtigen vor – das eine oder andere Tool wird aber sicherlich in einem neuen Licht erscheinen. Nützliche Tools sind meist nicht nur eine Arbeiterleichterung, sondern schaffen auch ungeahnte Möglichkeiten.

25.1.1 Der Rsync-Daemon

Während der *Rsync*-Client wohl für jeden Sysadmin vertrautes Terrain ist, ist der *Rsync*-Daemon (*RsyncD*) weniger populär. Dabei ist er eine erstklassige Alternative zu anderen Filetransferdiensten wie z. B. FTP. Im folgenden Beispiel konfigurieren Sie einen *Rsync*-Server, der das Verzeichnis `/srv/backup/` exportiert. In diesem Verzeichnis kann der Client später auf bequeme Weise Dateien sichern. Der *Rsync*-Daemon wird mit der Datei `rsyncd.conf` konfiguriert. Sie finden diese Datei je nach Distribution unter `/etc/` oder `/etc/rsync/`. Unter Umständen gibt es auch nur eine Beispieldatei im `doc`-Verzeichnis, die Sie nach `/etc/` kopieren und anpassen können. Eine funktionierende Konfigurationsdatei für unser Backupbeispiel sieht wie folgt aus:

```
read only = false
use chroot = true
transfer logging = true
log file = /var/log/rsyncd.log
hosts allow = 50.60.70.0/24
```

```
[backup]
path = /srv/backup
auth users = alice, bob, carl
secrets file = /etc/rsyncd.secrets
```

Listing 25.1 Konfigurationsdatei des Rsync-Daemons

Das bedeuten die einzelnen Einträge:

- ▶ **read only = false**
Die Clients dürfen nicht nur die Datei lesen, sondern auch schreiben.
- ▶ **use chroot = true**
Mit dieser Option wird verhindert, dass Clients aus der Verzeichnisstruktur des Rsync-Servers ausbrechen können.
- ▶ **transfer logging = true, log file = /var/log/rsyncd.log**
Die Zugriffe der Clients werden in der Datei `/var/log/rsyncd.log` protokolliert.
- ▶ **hosts allow = 50.60.70.0/24**
Es werden nur Zugriffe aus dem Netz 50.60.70.0/24 zugelassen.
- ▶ **[backup]**
Dies ist der Name der Verzeichnisfreigabe.
- ▶ **path = /srv/backup**
Hier wird der Pfad konfiguriert, in dem die Daten abgelegt werden, die die Clients hochladen.
- ▶ **auth users = alice, bob, carl**
Nur Benutzer aus dieser Liste dürfen die Freigabe nutzen.
- ▶ **secrets file = /etc/rsyncd.secrets**
In der Datei `/etc/rsyncd.secrets` stehen Paare aus Benutzername und Passwort, mit denen die Clients sich anmelden müssen. Der Inhalt der Datei kann etwa so aussehen:

```
alice:geheim
bob:meinpw
carl:nocheinpasswort
```

Listing 25.2 Inhalt: »rsyncd.secrets«

Wenn der Benutzer *bob* sein Homeverzeichnis auf dem Serverlaufwerk sichern möchte, gibt er dazu folgenden Befehl ein:

```
rsync -avz /home/bob/ bob@server.example.com::backup/bobshome/
```

Listing 25.3 Backup des Homeverzeichnisses auf einer Rsync-Freigabe

Entscheidend ist in Listing 25.3 die Adressierung mit zwei Doppelpunkten, da so der RsyncD angesprochen wird.

25.1.2 Wenn's mal wieder später wird: screen

Eine Terminalsitzung endet normalerweise, wenn Sie das Terminal schließen oder, falls Sie auf einem entfernten Server arbeiten, sobald Ihre SSH-Sitzung endet. Screen ist ein Terminal-Multiplexer, mit dem Sie das verhindern können. Rufen Sie einmal `screen` ohne weitere Parameter auf. Sie erhalten eine kurze Begrüßungsmeldung, die Sie mit einem Tastendruck bestätigen. Danach sehen Sie ein normal aussehendes Terminal. Rufen Sie nun den Systemmonitor `top` auf. Wenn er gestartet ist, schließen Sie das Terminal. Wenn Sie nun auf der Kommandozeile `screen -dr` eingeben, erhalten Sie Ihr eben geschlossenes Terminal zurück, und `top` läuft darin noch.

Wenn Sie mehrere persistente Terminalsitzungen haben möchten, können Sie innerhalb Ihrer Screen-Session weitere »Fenster«, also neue Terminalsitzungen, öffnen. Auch sie werden von Screen verwaltet. Das Öffnen, Schließen und Wechseln zwischen den Fenstern geschieht mit Tastenkombinationen, die alle mit `[Strg]+[A]` beginnen. In Tabelle 25.1 sehen Sie eine Liste der wichtigsten Kombinationen.

Kombination	Funktion	Bedeutung
<code>[Strg]+[A],["</code>	windows	gibt eine Liste aller offenen Fenster aus.
<code>[Strg]+[A],[C]</code>	create	erzeugt ein neues Fenster.
<code>[Strg]+[A],[N]</code>	next	wechselt zum nächsten Fenster.
<code>[Strg]+[A],[D]</code>	detach	hängt den Screen und die Session ab.
<code>[Strg]+[A],[K]</code>	kill	schließt das aktuelle Fenster.
<code>[Strg]+[A],[⇧]+[S]</code>	split	teilt das Terminalfenster horizontal (Splitscreen).
<code>[Strg]+[A],[⇧]+[↵]</code>	focus	wechselt den Fokus im Splitscreen.
<code>[Strg]+[A],[⇧]+[Q]</code>	only	hebt den Splitscreen und die Einteilung wieder auf.
<code>[Strg]+[A],[⇧]+[A]</code>	title	weist dem aktuellen Fenster einen Namen zu.
<code>[Strg]+[A],[⇧]+[?]</code>	help	zeigt alle Kombinationen und deren Funktion an.

Tabelle 25.1 Tastenkombinationen von »screen«

25.1.3 Anklopfen mit nmap

nmap (*network mapper*) ist ein Scanner zum Untersuchen von Netzwerken. Es ist das Standardwerkzeug für Portscans. Das Tool hat einen riesigen Funktionsumfang und kann sowohl für simple Portchecks als auch für groß angelegte Security-Audits verwendet werden. Im Fol-

genden werden die wichtigsten Funktionen zum Scannen der eigenen Infrastruktur gezeigt. Beim Aufruf von `nmap` müssen Sie das Ziel des Scans angeben. Das kann ein einzelner Host sein, ein Netz oder ein Adressbereich. Wenn außer dem Ziel keine weiteren Optionen angegeben werden, führt `nmap` einen *TCP-SYN-Scan*¹ (Option `-sS`) auf die 1000 am häufigsten verwendeten Ports durch. Bei einem SYN-Scan wird nur ein TCP-Paket mit gesetztem SYN-Flag gesendet. Das ist der erste Teil des sogenannten *Three-Way-Handshakes*. Danach antwortet der Server bei einem geöffneten Port mit einem SYN/ACK-Paket oder bei einem geschlossenen Port mit einem RST-Paket. Um den normalen Verbindungsaufbau fortzuführen und den Handshake abzuschließen, müsste `nmap` nun korrekterweise auf das SYN/ACK-Paket mit einem ACK antworten. Das tut es aber nicht, denn das Ziel ist bereits erreicht.

```
root@example:~# nmap scanme.nmap.org
Starting Nmap 7.80 ( https://nmap.org ) at 2023-01-03 13:49 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.18s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 993 closed ports
PORT      STATE  SERVICE
22/tcp    open   ssh
80/tcp    open   http
135/tcp   filtered msrpc
139/tcp   filtered netbios-ssn
445/tcp   filtered microsoft-ds
9929/tcp  open   nping-echo
31337/tcp open   Elite

Nmap done: 1 IP address (1 host up) scanned in 6.78 seconds
```

Listing 25.4 TCP-SYN-Scan mit »nmap«

Von den 1.000 gescannten Ports sind drei gefiltert, vier offen und die restlichen geschlossen. Als *filtered* werden alle Ports eingestuft, von denen entweder gar keine Antwort oder eine ICMP-Fehlermeldung kam. Näheres zu ICMP-Meldungen finden Sie in Abschnitt 22.6.7, »Das erste Firewallskript«.

Das Scannen von UDP-Ports ist nicht ganz so trivial, da es verbindungslos ist. Während sich TCP wie ein Einschreiben mit Rückschein verhält, wäre UDP eher eine Postkarte. Sie kann ankommen, muss es aber nicht. Normalerweise antworten offene UDP-Ports daher auch nur, wenn im UDP-Paket das Protokoll der Anwendung gesprochen wird. `nmap` schickt standardmäßig jedoch immer leere Pakete, weil es nicht wissen kann, welche Applikation hinter dem zu scannenden Port lauscht.

¹ Dazu muss man allerdings als *root*-Benutzer angemeldet sein. Normale Benutzer dürfen nur einen *TCP-Connect-Scan* durchführen.

Dementsprechend selten ist auch eine positive Antwort (*open*). Wenn *nmap* als Antwort jedoch eine *ICMP Port unreachable*-Nachricht erhält, bedeutet das, dass der Port eindeutig geschlossen ist (*closed*). Wenn andere ICMP-Fehlermeldungen kommen, wird das als gefilterter Port interpretiert (*filtered*).

Beim Ausbleiben einer Antwort wird *open|filtered* angezeigt, weil nicht eindeutig gesagt werden kann, ob das falsche Protokoll gesprochen wurde oder ob das Paket von einer Firewall verschluckt wurde. Um einen UDP-Scan durchzuführen, verwenden Sie die Option *-sU*:

```
root@example:~# nmap -sU 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2023-01-03 14:07 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000011s latency).
Not shown: 997 closed ports
PORT      STATE      SERVICE
68/udp    open|filtered dhcpc
161/udp   open       snmp
514/udp   open|filtered syslog

Nmap done: 1 IP address (1 host up) scanned in 1.28 seconds
```

Listing 25.5 UDP-Scan mit »nmap«

Etwas genauere Angaben über den Zustand der Ports erhalten Sie mit der Option *-sUV* (siehe Listing 25.6). Dieser Versionsscan probiert nämlich verschiedene Applikationsprotokolle auf den Ports aus, die als *open* und *open|filtered* gelten. Dadurch dauert der Scan aber auch sehr viel länger.

```
root@example:~# nmap -sUV 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2023-01-03 14:43 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000070s latency).
Not shown: 997 closed ports
PORT      STATE      SERVICE VERSION
68/udp    open|filtered dhcpc
161/udp   open       snmp     SNMPv1 server (public)
514/udp   open|filtered syslog

Service Info: Host: ubuntu

Service detection performed. Please report any incorrect results at \
http://nmap.org/submit/ .

Nmap done: 1 IP address (1 host up) scanned in 78.94 seconds
```

Listing 25.6 Detail-UDP-Scan mit »nmap«

Der Versionsscan ist übrigens auch mit TCP möglich. Damit kann man zum Teil herausfinden, welche Software hinter welchem Port arbeitet. Das funktioniert aber nicht mit einem *SYN-Scan*, sondern nur mit einem *TCP-Connect-Scan* (siehe Listing 25.7). Ohne einen vollständigen Verbindungsaufbau ist es schließlich nicht möglich, die Software direkt anzusprechen.

```
root@example:~# nmap -sV 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2023-01-03 15:01 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000050s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```
Service detection performed. Please report any incorrect results at [...]
Nmap done: 1 IP address (1 host up) scanned in 6.48 seconds
```

Listing 25.7 TCP-Connect-Scan mit »nmap«

Um die Erreichbarkeit Ihrer Dienste zu testen oder Ihre Firewallkonfiguration zu überprüfen, brauchen Sie noch weitere Optionen:

- ▶ **-p**
Mit `-p` können Sie Ports und Portbereiche angeben, die überprüft werden sollen. Beispiel:
`nmap -p 22,25,100-150 scanme.nmap.org`
- ▶ **-sP**
Diese Option führt einen *Ping-Scan* aus. Es wird also nicht geprüft, welche Ports offen sind, sondern lediglich, ob der Host online ist. Das ist sehr nützlich, um alle erreichbaren Rechner im lokalen Netz zu ermitteln. Beispiel: `nmap -sP 192.168.2.0/24`
- ▶ **-PN**
Normalerweise prüft `nmap` vor dem eigentlichen Portscan, ob der Host überhaupt online ist. Das wird mit einem Ping überprüft. Wenn der Host aber wegen einer Firewall nicht auf Pings reagiert, kann diese Prüfung mit der Option `-PN` deaktiviert werden.
- ▶ **-T**
Mit dieser Option kann ein *Timing Template* für den Scan ausgewählt werden. Ohne Angabe des Parameters wird `-T 3` (normal) genutzt. Bei sehr schnellen lokalen Netzen können Sie aber auch schnellere Scans durchführen. Erlaubte Werte sind `0` (*sneaky*) bis `5` (*insane*).
- ▶ **-O**
Diese Option aktiviert die *OS-Detection*. Anhand des Antwortverhaltens auf die Scans probiert `nmap`, das benutzte Betriebssystem des Zielrechners zu erraten. Die Ergebnisse sind allerdings nicht absolut zuverlässig.

► -F

Der *Fast Mode* scannt nur die 100 gebräuchlichsten Ports und nicht 1000, wie es in der Voreinstellung festgelegt ist.

Neben den genannten Optionen bietet *nmap* noch Dutzende weitere. Ein Blick in die Manpage lohnt sich auf jeden Fall. Viele Einstellungsmöglichkeiten setzen aber ein fundiertes Netzwerkwissen voraus.

25.1.4 Netzwerkinspektion: netstat

Auf einen Blick sehen, auf welchem Interface ein Dienst lauscht, welche Ports geöffnet sind oder mit welchem System eine Netzwerkverbindung besteht – all das bietet *netstat* (*NETwork STATistic*). Das Programm listet die aktuellen Netzwerkinformationen Ihres Systems tabellarisch gegliedert auf. Der wohl bekannteste Aufruf, allein schon aufgrund der sprechenden Parameterart, lautet `netstat -tulpen`. Mittlerweile ist *netstat* aber abgekündigt. Der würdige Nachfolger hört auf den unschönen Namen *ss* (*socket statistics*). Er gibt Ihnen die gleichen Informationen aus und versteht (überwiegend) die gleichen Parameter. Tabelle 25.2 zeigt die Bedeutung der Parameter `-tulpen` von *netstat* und deren Pendant bei *ss*, und Listing 25.8 zeigt eine Beispielausgabe:

```
root@ubuntu:~# ss -tulpen
```

```
Netid State Recv-Q Send-Q Local Address:Port      Peer Address:Port
udp   UNCONN 23040 0       127.0.0.53%lo:53      0.0.0.0:* users:(("syst[...]
udp   UNCONN 0      0       10.0.0.6%enp0s3:68    0.0.0.0:* users:(("syst[...]
tcp   LISTEN 0      128     127.0.0.53%lo:53      0.0.0.0:* users:(("syst[...]
tcp   LISTEN 0      128     0.0.0.0:22           0.0.0.0:* users:(("sshd[...]
tcp   ESTAB  0      0       10.0.0.6:22          10.0.0.3:64155 users:(("sshd[...]
tcp   LISTEN 0      128     [::]:22              [::]:* users:(("sshd[...]
[...]
```

Listing 25.8 Übersicht der Netzwerkstatistiken mit »ss« (Pendant zu »netstat -tulpen«)

netstat / ss	Bedeutung
-t ---tcp	zeigt TCP-Protokolle.
-u ---udp	zeigt UDP-Protokolle.
-l ---listening	sorgt für die Darstellung von lauschenden Sockets.
-p ---processes	zeigt auch die PID und den Programmnamen.

Tabelle 25.2 »netstat«- und »ss«-Parameter

netstat / ss	Bedeutung
-e ---extended	gibt zusätzliche Informationen zur User ID und zum Inode mit aus.
-n ---numeric	unterbindet die Namensauflösung (deutlich schnellere Ausgabe).

Tabelle 25.2 »netstat«-Parameter (Forts.)

Eine so erzeugte Ausgabe enthält in der tabellarischen Auflistung folgende Informationen:

- ▶ **Proto**
Protokoll (TCP, UDP, ICMP etc.)
- ▶ **Recv-Q**
die Anzahl von Bytes, die die Anwendung noch nicht vom Socket abgeholt hat
- ▶ **Send-Q**
die Anzahl von Bytes, die von der Gegenseite noch nicht bestätigt wurden
- ▶ **Local Address**
lokale Adresse
- ▶ **Foreign Address**
entfernte Adresse
- ▶ **State**
Status der Verbindung; dabei sind folgende Status möglich:
 - **ESTABLISHED**
bestehende Verbindung
 - **SYN_SENT**
Der Verbindungsaufbau auf dem Socket ist im Gange.
 - **SYN_RECV**
Die Verbindungsanfrage wurde von der Gegenseite empfangen.
 - **FIN_WAIT1**
Der Socket wurde geschlossen, und die Verbindung wird beendet.
 - **FIN_WAIT2**
Die Verbindung ist geschlossen – Warten auf die Beendigung der Gegenseite.
 - **CLOSE**
Der Socket wird nicht benutzt.
 - **CLOSE_WAIT**
Die Gegenseite hat die Verbindung beendet – das Schließen des Sockets wird erwartet.
 - **TIME_WAIT**
Der Socket ist geschlossen und im Wartezustand (noch nicht beendete Verbindung).

- **LAST_ACK**
Die Gegenseite hat die Verbindung beendet, und der Socket ist geschlossen. Eine abschließende Bestätigung wird abgewartet.
 - **LISTEN**
Der Socket wartet auf eingehende Verbindungen.
 - **CLOSING**
Beide Sockets (der lokale und der entfernte) sind geschlossen. Es wurden aber noch nicht alle Daten geschickt.
 - **UNKNOWN**
Der Zustand des Sockets ist unbekannt.
- ▶ **User**
Eigentümer des Sockets
 - ▶ **Inode**
Inode-Nummer des Sockets
 - ▶ **PID/Program name**
die PID und der Programmname des Programms, das den Socket geöffnet hat

Die Programme `netstat` und `ss` verfügen noch über viele weitere Parameter, die Sie den Manpages und den Infopages entnehmen können.

25.1.5 Zugreifende Prozesse finden: `lsof`

`lsof` steht für *list open files* und macht genau das, was der Name verspricht: Es zeigt an, welche Prozesse auf welche Dateien zugreifen. Da unter Linux so gut wie alles eine Datei ist, werden auch *Sockets*, *Pipes*, *Netzwerkressourcen* und vieles mehr angezeigt. Mit `lsof` lässt sich so sehr einfach ermitteln, welcher Prozess gerade welche Ressourcen benutzt. Ruft man `lsof` ohne Parameter auf, so werden alle zurzeit benutzten Dateien angezeigt. Bei einem normalen Linux-System sind das eine ganze Menge. Mit verschiedenen Optionen können Sie aber auch gezielt nach Informationen suchen.

Prozesssuche: `lsof -p PID`

Gibt alle geöffneten Ressourcen des Prozesses mit der angegebenen PID aus.

```
root@example:~# lsof -p 1
COMMAND PID USER  FD   TYPE    DEVICE  SIZE/OFF      NODE NAME
systemd  1 root  cwd   DIR     252,0    4096         2 /
systemd  1 root  rtd   DIR     252,0    4096         2 /
systemd  1 root  txt   REG     252,0   1573136   921775  /lib/systemd/systemd
[...]
```

Listing 25.9 Ausgabe für den Prozess mit der ID »1«

Netzwerkressourcen: lsof -i

Alle genutzten Netzwerkressourcen können Sie mit `lsof -i` anzeigen lassen. Über die zusätzliche Angabe eines Protokolls, Hostnamens und Ports kann die Anzeige eingeschränkt werden. Um beispielsweise alle IMAP-Verbindungen zu sehen, können Sie `lsof -i tcp:143` aufrufen. Über die Optionen `-Pn` können Sie das Auflösen von Port- und Hostnamen unterbinden:

```
root@example:~# lsof -Pni
COMMAND  PID    USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
dhclient3 798   root   4u   IPv4  7033    0t0   UDP *:68
sshd     833   root   3u   IPv4  7469    0t0   TCP *:22 (LISTEN)
sshd     833   root   4u   IPv6  7471    0t0   TCP *:22 (LISTEN)
sshd     849   root   3r   IPv4  7491    0t0   TCP 192.168.2.16:22\
->192.168.2.10:39975 (ESTABLISHED)
[...]
```

Listing 25.10 Ausgabe der aktiven Netzwerkverbindungen

Dateizugriffe: lsof /path/file

Wenn Sie `lsof`, gefolgt von einer Datei oder einem Verzeichnis, aufrufen, werden Ihnen alle Prozesse angezeigt, die aktuell darauf zugreifen. Angewendet auf einen Mountpoint, erhalten Sie die Ausgabe aller geöffneten Dateien auf dem Dateisystem.

```
root@example:~# lsof /bin/bash
COMMAND  PID  USER  FD  TYPE DEVICE SIZE/OFF NODE NAME
bash     929 daniel txt  REG 253,0 1396520 516 /usr/bin/bash
bash     949 root  txt  REG 253,0 1396520 516 /usr/bin/bash
```

Listing 25.11 Ausgabe der Zugriffe auf die Datei »/bin/bash«

Die einzelnen Spalten der Ausgabe werden wir uns am Beispiel der geöffneten Dateien des `rsyslog`-Daemons ansehen:

```
root@example:~# lsof -p 5441
COMMAND  PID  USER  FD  TYPE  DEVICE  SIZE/OFF      NODE  NAME
rsyslogd 5441 syslog  cwd  DIR    253,0    4096           2     /
rsyslogd 5441 syslog  rtd  DIR    253,0    4096           2     /
rsyslogd 5441 syslog  txt  REG    253,0   287928      8294  /usr\
/sbin/rsyslogd
rsyslogd 5441 syslog  mem  REG    253,0   88384        857  /lib\
/libgcc_s.so.1
rsyslogd 5441 syslog  mem  REG    253,0   43552       1005  /lib\
/libnss_nis-2.12.1.so
[...]
```

```
rsyslogd 5441 syslog  0u  unix    0x...      0t0      123319 /dev/log
```



```

rsyslogd 5441 syslog 3u IPv4 123313 0t0 UDP *:syslog
rsyslogd 5441 syslog 4u IPv6 123314 0t0 UDP *:syslog
rsyslogd 5441 syslog 5r REG 0,3 0 4026532018 /proc/kmsg
[...]
rsyslogd 5441 syslog 8w REG 253,0 2691 138857 /var/log\
/syslog
[...]
```

Listing 25.12 Beispiel: Zugriffe des »rsyslogd«

Dabei haben die Spalten nachstehende Bedeutung:

- ▶ **COMMAND**
der Name des Prozesses, der auf die angegebene Ressource zugreift
- ▶ **PID**
die Prozess-ID des Prozesses
- ▶ **USER**
der Benutzername, unter dem der Prozess läuft
- ▶ **FD**
Mit FD wird entweder die Nummer des *File Descriptors* der Ressource oder der Ressourcen-typ angegeben. Die wichtigsten Werte für FD sind folgende:
 - **cwd**
Mit *cwd* (*current working directory*) wird das aktuelle Arbeitsverzeichnis des Prozesses ausgegeben, also in der Regel das Verzeichnis, aus dem heraus der Prozess gestartet wurde.
 - **rtd**
gibt das Wurzelverzeichnis (*root directory*) des Prozesses an. Bei den meisten Prozessen ist das /, in einer chroot-Umgebung kann es aber auch ein beliebiges anderes Verzeichnis sein.
 - **txt**
steht für *program text* und zeigt das ausgeführte Programm an.
 - **mem**
Der Typ *mem* zeigt an, dass die angegebene Datei in den Speicher »gemappt« wurde. Das geschieht in der Regel bei Bibliotheken.
 - **Ou, 5r, 8w ...**
Bei Zahlen, auf die die Buchstaben *u*, *r* oder *w* folgen, handelt es sich um die Nummern des *File Descriptors* (FD). Beim Öffnen von Dateien wird der Datei ein für den Prozess eindeutiger FD zugewiesen. Alle weiteren Operationen (wie das Lesen, Schreiben und Schließen der Datei) erfolgen dann nur über die Nummer des FD und nicht über den Namen. Diese Nummern kennen Sie schon von der Shell. Die File Descriptors 0, 1 und

2 Ihrer Shell sind *STDIN*, *STDOUT* und *STDERR*. Mit der Umleitung `KOMMANDO 2> DATEI` biegen Sie letztendlich das Ziel des FDs 2 um. Die Buchstaben nach dem File Descriptor geben den Zugriffsmodus an: *r* steht für lesenden Zugriff, *w* für schreibenden und *u* für beides. In Listing 25.12 sehen Sie beispielsweise, dass */var/log/syslog* gerade für einen schreibenden Zugriff unter der Nummer 8 zur Verfügung steht.

► **TYPE**

gibt die Art der Datei an. Es gibt sehr viele verschiedene Typen. Die wichtigsten sind *REG* für eine reguläre Datei, *DIR* für ein Verzeichnis, *CHR* und *BLK* für Character- bzw. Block-Devices und *IPV4/6* für geöffnete IP-Verbindungen.

► **DEVICE**

Device enthält bei »normalen« Dateien die *Major* und *Minor Number* des Geräts. Die Datei */usr/sbin/rsyslogd* aus dem Beispiel in Listing 25.12 liegt auf dem Gerät 253,0. Das entspricht */dev/md2*. Neben der Gerätenummer kann die *DEVICE*-Spalte auch eine Referenznummer des Kernels enthalten, unter der das Gerät adressiert wird.

► **SIZE/OFF**

In dieser Spalte wird die Größe der Datei angezeigt. Einträge, die mit *0t* oder *0x* beginnen, zeigen den Offset an.

► **NODE**

Unter Node finden Sie entweder die *Inode-Nummer* der Datei oder den verwendeten Protokolltyp, wie beispielsweise *TCP* oder *UDP*.

► **NAME**

Die letzte Spalte gibt den Dateinamen an oder bei Netzwerkressourcen die lokale sowie gegebenenfalls entfernte Adresse und den Verbindungszustand.

Mit *lsOf* lassen sich viele Fragestellungen rund um belegte Ressourcen leicht beantworten. Um beispielsweise herauszufinden, welcher Prozess das Aushängen eines Dateisystems blockiert, reicht der Aufruf von `lsOf /mountpoint`. Sehr praktisch ist auch die Möglichkeit, auf gelöschte, aber noch aktive Dateien zuzugreifen. Solange ein Prozess eine Datei noch geöffnet hat, ist sie über den File Descriptor nämlich noch zu erreichen, auch wenn sie im Dateisystem nicht mehr zu sehen ist. Wenn Sie beispielsweise die Log-Datei */var/log/syslog* löschen, während der *Syslog*-Daemon noch läuft, erhalten Sie folgende Ausgabe von *lsOf*:

```
root@example:~# rm /var/log/syslog
root@example:~# ls /var/log/syslog
ls: Zugriff auf /var/log/syslog nicht möglich: No such file or directory

root@example:~# lsOf -p 5441
[...]
rsyslogd 5441 syslog 8w REG 253,0 2786 138857 /var/log/syslog (deleted)
```

Listing 25.13 Zugriffe auf gelöschte Dateien

Über den FD 8 und mithilfe des *proc*-Dateisystems können Sie die Daten aber noch einsehen und bei Bedarf wiederherstellen (Näheres zum *proc*-Dateisystem finden Sie in Abschnitt 3.4, »Alles virtuell? ›/proc«):

```
root@example:~# cat /proc/5441/fd/8
```

Listing 25.14 Ausgabe der gelöschten Dateien

25.1.6 Was macht mein System? top

top ist eines der Standardwerkzeuge, wenn es darum geht, den aktuellen Systemzustand zu prüfen. Bei den meisten Administratoren, die wir kennen, ist es das erste Kommando, das ausgeführt wird, wenn es auf einer Maschine zu Problemen kommt. Das Programm zeigt in den ersten Zeilen einige allgemeine Informationen über den Systemzustand an. Der größte Teil des Bildschirms wird aber für die Anzeige der Prozesse genutzt, die aktuell die meiste Rechenzeit verbrauchen. Die Ansicht aktualisiert sich in der Standardeinstellung alle drei Sekunden selbst.

Die Ausgabe von *top* lässt sich relativ frei konfigurieren. So lassen sich beispielsweise zusätzliche Spalten mit Detailinformationen zu den Prozessen ein- und ausblenden. Auch die Sortierreihenfolge kann geändert werden, um beispielsweise auf einen Blick die größten Speichersünder zu identifizieren.

```
top - 10:03:57 up 4 days, 11:31, 2 users, load average: 2.36, 2.14, 1.86
Tasks: 179 total, 3 running, 176 sleeping, 0 stopped, 0 zombie
%Cpu(s): 23.9 us, 6.4 sy, 0.0 ni, 63.1 id, 6.1 wa, 0.1 hi, 0.3 si, 0.0 st
MiB Mem : 4838500k total, 48224k free, 4790276k used, 117576k buff/cache
MiB Swap: 1951888k total, 1903576k free, 48312k used, 1733664k avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1711	www-data	16	0	282m	44m	5140	S	31	0.9	0:03.75	apache2
1537	www-data	15	0	291m	52m	7124	S	14	1.1	0:01.31	apache2
1622	www-data	16	0	289m	49m	7632	S	11	1.0	0:01.85	apache2
1162	www-data	15	0	311m	73m	7496	S	11	1.6	0:05.13	apache2
1168	www-data	15	0	296m	57m	7400	S	10	1.2	0:02.98	apache2
1718	www-data	15	0	299m	63m	5316	R	9	1.3	0:00.82	apache2

Listing 25.15 Die Ausgabe von »top«

top bietet eine ganze Reihe von Kommandozeilenoptionen an:

► **-b**

aktiviert den Batchmodus. *top* startet dann im nicht interaktiven Modus und gibt im Standardintervall Prozessinformationen aus. Der Batchmodus eignet sich hervorragend, um die Ausgabe von anderen Programmen weiterverarbeiten zu lassen.

- ▶ **-n <ZAHL>**
gibt die Anzahl der Updates an, die ausgeführt werden sollen, bevor sich `top` beendet. Die Option ist in Kombination mit dem Batchmodus am sinnvollsten.
- ▶ **-u <USER>**
zeigt nur Prozesse vom angegebenen Benutzer an. `top -u www-data` zeigt beispielsweise nur die Prozesse des Webservers an.
- ▶ **-d <SEKUNDE>.<ZEHNTTEL>**
setzt das Aktualisierungsintervall.
- ▶ **-p <PID1>,<PID2>,<...>**
zeigt nur ausgewählte Prozesse an.

Die wichtigsten Informationen in den oberen Zeilen von Listing 25.15 sind folgende:

- ▶ **load average**
gibt die durchschnittliche Systemlast an. Die drei Zahlen sind ein exponentiell geglätteter Wert, der die durchschnittliche Anzahl von Prozessen in der *run queue* über einen Zeitraum von 1, 5 und 15 Minuten angibt. Das heißt, es wird angezeigt, wie viele Prozesse zurzeit laufen oder gerne laufen würden. Die *load* ist übrigens ein Wert, der äußerst subjektiv bewertet werden muss. Ein System mit einer *load* von 3 muss nicht langsamer oder überlasteter sein als eines mit einer *load* von 1,8. Sie müssen selbst sehen, ab wann ein System zu träge reagiert, und so die maximale *load* ermitteln. Abhängig von der Anzahl der CPU-Kerne und der Aufgabe des Systems kann das sehr unterschiedlich sein. Das Auswerten der *load average* ist aber dennoch sinnvoll: Durch das Monitoring dieses Werts können Sie Lastspitzen leicht erkennen und Kapazitätsplanung betreiben.
- ▶ **us**
gibt an, wie viel Prozent der Rechenzeit für Benutzerprozesse verwendet wird. Auf einem ausgelasteten System ist dieser Wert sehr hoch.
- ▶ **sy**
zeigt die benutzte Zeit in Prozent, die der Kernel selbst für interne Aufgaben benutzt hat. Wenn die *system time* sehr hoch ist, ist das oft ein Hinweis auf eine Fehlfunktion. Die *user time* sollte normalerweise um ein Vielfaches höher sein als die *system time*. Auf nicht korrekt funktionierenden Virtualisierungsumgebungen sieht man häufig den umgekehrten Fall, beispielsweise bei Linux-Gästen in einer VMware-Umgebung, die ohne die VMware-Tools laufen. Eine hohe *system time* sagt aus, dass sich das System mehr mit sich selbst als mit den eigentlichen Aufgaben beschäftigt.
- ▶ **ni**
ist die Zeit, die *genicete* Prozesse verbraucht haben, also *user time*-Prozesse, die höher oder niedriger priorisiert sind.
- ▶ **id**
zeigt an, wie viel Prozent der Zeit das System nichts tut.

- ▶ **wa**
gibt an, wie lange anteilig auf das Fertigstellen von I/O-Operationen gewartet werden muss. Dieser Wert sollte möglichst gering sein. Ein hohes *iowait* bedeutet meistens, dass das Storage-System nicht mit den nötigen Lese- und Schreiboperationen mithalten kann.
- ▶ **hi**
verbrauchte Zeit für das Ausführen von Hardware-Interrupts
- ▶ **si**
verbrauchte Zeit für das Ausführen von Software-Interrupts
- ▶ **Mem**
In dieser Zeile sieht man, wie viel RAM insgesamt zur Verfügung steht, wie viel verbraucht ist und wie viel noch übrig ist. Der Wert für den freien Speicher muss mit Vorsicht genossen werden. Denn alles, was nicht direkt für Applikationen genutzt wird, benutzt der Kernel für das Caching. Bei Bedarf kann der Kernel diese Speicherbereiche wieder freigeben. Man muss daher die Werte von *cached* und *buffers* noch zu dem freien Speicher addieren, um einen realistischeren Wert zu bekommen. `free -m` zeigt genau das an. Mit `slabtop` kann man sich übrigens anzeigen lassen, wofür der Cache genutzt wird.
- ▶ **Swap**
zeigt an, wie groß der Swap-Bereich ist und wie viel belegt ist.

Die angezeigten Spalten in der Prozessübersicht können frei gewählt werden. Durch Drücken von können zusätzliche Spalten ein- und ausgeblendet werden. Hier sehen Sie die wichtigsten Felder im Überblick:

- ▶ **PID**
die Prozess-ID. Bei Bedarf kann auch die PPID (*Parent Process PID*) eingeblendet werden.
- ▶ **USER**
der effektive Benutzer des Prozesses. Alternativ kann auch der echte angezeigt werden.
- ▶ **PR**
gibt die Priorität des Tasks an. Je höher der Wert ist, desto länger hat der Prozess bereits auf die Zuteilung von Rechenzeit gewartet.
- ▶ **NI**
zeigt den Nice-Wert an. Je höher die Zahl ist, desto geringer ist die Priorität.
- ▶ **VIRT**
steht für *virtual image* und sagt aus, auf wie viel Speicher der Prozess aktuell Zugriff hat. Dieser Wert beinhaltet auch ausgelagerten Speicher im Swap-Bereich und *shared libraries*.
- ▶ **RES**
heißt *resident image* und gibt im Gegensatz zu dem *virtual image* den realen physischen Speicherverbrauch an, allerdings ohne *shared libraries*. Dieser Wert ist am wichtigsten, um den Speicherverbrauch von Prozessen zu analysieren.

► SHR

ist die *shared memory size* und gibt an, wie viel RAM für *shared libraries* genutzt wird. Dieser Speicher kann mit anderen Prozessen geteilt werden.

► S

gibt Aufschluss über den Prozesszustand:

- **D**
uninterruptable sleep – der Prozess schläft und kann nicht aufgeweckt werden.
- **R**
ein laufender oder lauffähiger Prozess
- **S**
Der Prozess schläft.
- **T**
traced or stopped – der Prozess wurde durch ein Signal angehalten.
- **Z**
zombie – der Prozess ist tot, wurde vom Elternprozess jedoch nicht sauber beendet.

► %CPU

zeigt an, wie viel Prozent der zur Verfügung stehenden Rechenleistung gerade von dem Prozess verwendet wird.

► %MEM

zeigt den prozentualen Speicherverbrauch des Prozesses an.

► TIME+

gibt an, wie viel CPU-Zeit der Prozess schon verbraucht hat.

► COMMAND

zeigt das ausgeführte Kommando an.

Mit »iotop« die Festplattenaktivität analysieren

Wenn die Festplatten in hektische Aktivität ausbrechen, ist es oft nicht einfach, den dafür verantwortlichen Prozess ausfindig zu machen. *iotop* ist dabei eine große Hilfe. Wenn Sie *iotop* ohne weitere Parameter aufrufen, präsentiert es Ihnen in der Kopfzeile der Ausgabe die kumulierten Datenraten aller schreibenden und lesenden Zugriffe. Darunter sehen Sie eine Prozesstabelle, in der Sie die Zugriffsraten pro Prozess ablesen können (siehe Listing 25.16). So finden Sie schnell und einfach einen Prozess, der durch außergewöhnlich häufige Festplattenzugriffe auffällt.

```
Total DISK READ :      0.00 B/s | Total DISK WRITE :      5.76 M/s
Actual DISK READ:      0.00 B/s | Actual DISK WRITE:      0.00 B/s
  TID  PRIO  USER      DISK READ  DISK WRITE  SWAPIN     IO>   COMMAND
 5322 be/4  root        0.00 B/s   0.00 B/s   0.00 %    0.02 % [kworker/u2:1]
 5687 be/4  daniel      0.00 B/s   5.76 M/s   0.00 %    0.00 % dd if=/dev/[...]
```

```

1 be/4 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % init
2 be/4 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % [kthreadd]
3 be/4 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % [ksoftirqd/0]
5 be/0 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % [kworker/0:0H]
7 be/4 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % [rcu_sched]
8 be/4 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % [rcu_bh]
9 rt/4 root      0.00 B/s   0.00 B/s  0.00 %  0.00 % [migration/0]
10 rt/4 root     0.00 B/s   0.00 B/s  0.00 %  0.00 % [watchdog/0]
11 be/4 root     0.00 B/s   0.00 B/s  0.00 %  0.00 % [kdevtmpfs]
12 be/0 root     0.00 B/s   0.00 B/s  0.00 %  0.00 % [netns]
13 be/0 root     0.00 B/s   0.00 B/s  0.00 %  0.00 % [perf]
14 be/4 root     0.00 B/s   0.00 B/s  0.00 %  0.00 % [khungtaskd]
15 be/0 root     0.00 B/s   0.00 B/s  0.00 %  0.00 % [writeback]

```

Listing 25.16 Ausgabe von »iotop« (Auszug)

iotop erlaubt es zusätzlich, die Datenraten in Intervallen zu messen und in eine Datei zu schreiben. So können Sie die Entwicklung der Plattenaktivität über einen längeren Zeitraum beobachten. Dazu starten Sie *iotop* zum Beispiel mit dem folgenden Befehl:

```
iotop -o -b -d10 -n100 > hdd.txt
```

Listing 25.17 Langzeitmessung der Plattenaktivität mit »iotop«

iotop schreibt die Messwerte nun einhundert Mal, jeweils im Abstand von zehn Sekunden, in die Datei *hdd.txt*.

25.1.7 Wenn gar nichts mehr geht – Debugging mit *strace*

strace ist ein ziemliches Schwergewicht unter den Analysetools und manchmal die letzte Möglichkeit, um herauszufinden, was ein Prozess gerade tut oder woran er scheitert. Mithilfe von *strace* lassen sich alle Systemaufrufe (*system calls*) ausgeben, die ein Prozess gerade durchführt. Ein Systemaufruf ist eine Anweisung an den Kernel, um auf Ressourcen zuzugreifen. Aus Sicherheitsgründen darf kein Prozess direkt Dateien öffnen, schließen oder Speicheradressen manipulieren. Letztendlich wird also alles außer prozessinternen Berechnungen über Systemaufrufe erledigt. Dementsprechend umfangreich ist auch die Ausgabe von *strace*. Eine Liste von *system calls* finden Sie in der Manpage von *syscalls*. Jeder Systemaufruf ist wiederum in einer eigenen Manpage dokumentiert.

Es gibt sehr viele *system calls*, und viele kann man als Nicht-Programmierer kaum verstehen. In Kombination mit der riesigen Informationsflut, die selbst bei kleinen Programmen durch *strace* ausgegeben wird, wird die Fehleranalyse schnell zur Suche nach der Nadel im Heuhaufen. Mit etwas Übung und Geduld kann man durchaus zu einem brauchbaren Ergebnis kommen.



Es gibt zwei Möglichkeiten, wie Sie *strace* einsetzen können. Bei der ersten Variante, *strace* PROGRAMM, wird das Programm direkt gestartet und sofort »getracet« (es wird also der Spur des Programms gefolgt). Wenn Sie einen bereits laufenden Prozess untersuchen möchten, dann können Sie *strace -p* PID aufrufen. In diesem Fall dockt sich *strace* an den Prozess an und gibt fortlaufend die aufgerufenen *system calls* aus. Weitere nützliche Optionen sind *-o* DATEI und *-f*. Mit *-o* können Sie die Ausgabe von *strace* in eine Datei schreiben lassen. Die Option *-f* ist notwendig, um auch Unterprozesse zu verfolgen. Sehr hilfreich ist die Option *-e*. Mit ihr können Sie die Ausgabe von *strace* auf eine Liste von *system calls* beschränken oder gezielt einige ausschließen. *strace -e open,read,write ...* gibt nur *open*-, *read*- und *write*-*system calls* aus. *strace -e \!mmap,brk,mprotect* würde alle *system calls* außer den aufgezählten ausgeben.

Im Folgenden werden wir *strace* am Beispiel des Befehls *true* untersuchen. Der einzige Zweck des Kommandos ist es, den *Exit-Status 0* zurückzugeben. Es ist damit eines der einfachsten Programme überhaupt.

```
root@example:~# strace true
execve("/usr/bin/true", ["true"], 0x7ffdb12bac00 /* 21 vars */) = 0
brk(NULL) = 0x55fa9a68a000
arch_prctl(0x3001 /* ARCH_??? */ , 0x7ffc32852030) = -1 EINVAL (Invalid argument)
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
open(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, st_mode=S_IFREG|0644, st_size=24424, ...) = 0
mmap(NULL, 24424, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f643e15e000
close(3) = 0
open(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360q\2\0\0\0\0"... , \
832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@0\0\0\0\0\0@0\0\0\0\0\0@0\0\0\0\0\0"... , 784, \
64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0", 32, \
848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355\ [...]"
fstat(3, st_mode=S_IFREG|0755, st_size=2029224, ...) = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) \
= 0x7f643e15c000
pread64(3, "\6\0\0\0\4\0\0\0@0\0\0\0\0\0@0\0\0\0\0\0@0\0\0\0\0\0"... , 784, \
64) = 784
pread64(3, "\4\0\0\0\20\0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0", 32, \
848) = 32
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0\363\377?\332\200\270\27\304d\245n\355Y\ [...]"
mmap(NULL, 2036952, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7f643df6a000
mprotect(0x7f643df8f000, 1847296, PROT_NONE) = 0
```



```

mmap(0x7f643df8f000, 1540096, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED| [...]
mmap(0x7f643e107000, 303104, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, [...]
mmap(0x7f643e152000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED| [...]
mmap(0x7f643e158000, 13528, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED| [...]
close(3) = 0
arch_prctl(ARCH_SET_FS, 0x7f643e15d580) = 0
mprotect(0x7f643e152000, 12288, PROT_READ) = 0
mprotect(0x55fa9888e000, 4096, PROT_READ) = 0
mprotect(0x7f643e191000, 4096, PROT_READ) = 0
munmap(0x7f643e15e000, 24424) = 0
exit_group(0) = ?
+++ exited with 0 +++

```

Listing 25.18 Ausgabe der Aktionen des Programms »true« durch »strace«

Wie Sie sehen, werden für eine eigentlich recht simple Aufgabe sehr viele *system calls* ausgeführt. Sie können sich also leicht vorstellen, was beim Tracen eines Webserver ausgegeben wird. Zum Glück sind für die meisten Analysen aber nur wenige Systemaufrufe wirklich relevant. Die wichtigsten Aufrufe aus Listing 25.18 werden im Folgenden beschrieben:

► **execve**

Dieser Systemaufruf führt das Programm mit dem nachfolgenden Namen aus. Zusätzlich sehen Sie die Aufrufparameter und die Anzahl der für den Prozess gesetzten Umgebungsvariablen. Sie können sich auch die Variablen selbst anzeigen lassen, indem Sie *strace* zusätzlich mit der Option *-v* aufrufen. Der letzte Eintrag jeder Zeile ist der Rückgabewert des Systemaufrufs. Negative Werte zeigen einen Fehler an.

► **access**

Der Systemaufruf *access* prüft, ob auf eine Datei mit den gewünschten Rechten (Lesen, Schreiben etc.) zugegriffen werden kann. In Listing 25.18 können Sie sehen, dass auf */etc/ld.so.preload* nicht zugegriffen werden kann, da die Datei nicht existiert, was mit einer Fehlermeldung quittiert wird: *(-1 ENOENT (No such file or directory))*.

► **open**

open dient zum Öffnen von Dateien. Der Rückgabewert des Systemaufrufs ist der zugewiesene File Descriptor. Beim *open*-Aufruf wird mit angegeben, wie auf die Datei zugegriffen werden soll. Im Beispiel wird */etc/ld.so.cache* lesend (*O_RDONLY*) geöffnet und ist fortan mit dem File Descriptor 3 ansprechbar.

► **read**

Mit *read* werden Daten aus dem angegebenen File Descriptor gelesen. Die ersten gelesenen 32 Bytes werden mit ausgegeben. Wenn Sie mehr Daten angezeigt haben möchten, können Sie mit der Option *-s* eine andere Größe setzen.

► **fstat**

Gibt Informationen über die Datei mit dem angegebenen File Descriptor zurück.

► **close**

Geöffnete Dateien werden mit `close` wieder geschlossen. Als Argument wird der File Descriptor angegeben.

► **exit_group**

Dieser *system call* liefert endlich das gewünschte Ergebnis. Er gibt uns den Exit-Status 0 zurück.

Um ein Gefühl für einen typischen Programmablauf zu bekommen und die Systemaufrufe näher kennenzulernen, sollten Sie einfach ein paar Dienste und einfache Tools *tracen*. Das folgende Beispiel zeigt den *rsyslog*-Daemon bei der Arbeit.

Diesmal sollen nur *write*-Systemaufrufe protokolliert werden (`-e write`). Dabei sollen die ersten 1024 Byte mit ausgegeben werden (`-s 1024`) sowie sämtliche Unterprozesse (`-f`):

```
root@example:~# strace -s 1024 -f -e write -p 6151
[...]
[pid 6283] write(8, "Nov 3 19:20:40 example daniel: Hallo Leser\n", 43) = 43
[pid 6283] write(13, "Nov 3 19:20:40 example daniel: Hallo Leser\n", 43) = 43
[pid 6283] write(21, "Nov 3 19:20:40 example daniel: Hallo Leser\n", 43) = 43
[...]
```

Listing 25.19 »rsyslogd«-Systemaufrufe

Das Ergebnis zeigt, wie auf die *File Descriptors* 8, 13 und 21 jeweils die gleiche 43 Byte große Log-Meldung geschrieben wurde. Die FDs gehören zu den Dateien `/var/log/syslog`, `/var/log/user` und `/var/log/messages`.

Im nächsten Beispiel schauen wir uns einige Elemente einer FTP-Sitzung an. Bei größeren Programmen sollten Sie die Ausgabe mit der Option `-o` immer in eine Datei schreiben.

```
root@example:~# strace -o /tmp/ftp.trace ftp ftp.gwdg.de
[...]
```

Listing 25.20 »ftp«-Systemaufrufe

Der Aufruf erzeugt einen Output von mehr als 500 Zeilen. Zunächst betrachten wir nur einige *open*-Systemaufrufe:

```
[...]
open(AT_FDCWD, "/lib/x86_64-linux-gnu/libedit.so.2", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/lib/x86_64-linux-gnu/libssl.so.3", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/lib/x86_64-linux-gnu/libcrypto.so.3", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
[...]
open(AT_FDCWD, "/lib/x86_64-linux-gnu/glibc-hwcap/x86-64-v2/libnss_db.so.2", \
O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
```

```

open(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64/x86_64/libnss_db.so.2", \
  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
open(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64/libnss_db.so.2", \
  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory) \
open(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/x86_64/libnss_db.so.2", \
  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
open(AT_FDCWD, "/lib/x86_64-linux-gnu/tls/libnss_db.so.2", \
  O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
[...]
open(AT_FDCWD, "/etc/host.conf", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/etc/resolv.conf", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/etc/hosts", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/etc/gai.conf", O_RDONLY|O_CLOEXEC) = 3
open(AT_FDCWD, "/root/.netrc", O_RDONLY) = -1 ENOENT (No such file or directory)
[...]

```

Listing 25.21 Auszug aus »/tmp/ftp.trace«

In der Ausgabe von Listing 25.21 sehen Sie, wie zunächst einige wichtige Bibliotheken geladen werden. Sie können auch sehen, dass eine ganze Menge open-Operationen scheitern, weil die gewünschten Dateien einfach nicht existieren. Das ist völlig normal und muss nicht zwangsläufig ein kritischer Fehler sein. Beim Starten von Prozessen wird oft probiert, optionale Konfigurationsdateien oder Bibliotheken zu laden. Es ist manchmal schwierig, zu beurteilen, ob das Fehlen einer Datei problematisch ist oder nicht. Aus dem Dateinamen kann man aber erste Schlüsse zu Funktion und Wichtigkeit ziehen. Der letzte Block des Listings zeigt eine typische Auswahl an Komponenten, die für die Namensauflösung gebraucht werden. Mit der Zeit werden Sie feststellen, dass sich bei fast jedem Programmaufruf die gleichen Schritte immer wiederholen.

Interessanter wird der Netzwerkteil:

```

socket(PF_INET, SOCK_STREAM, IPPROTO_IP) = 3
[...]
connect(3, {sa_family=AF_INET, sin_port=htons(21), \
  sin_addr=inet_addr("134.76.12.3")}, 16) = 0
[...]
dup(3) = 5
[...]

```

Listing 25.22 Fortsetzung von »/tmp/ftp.trace«

Mit socket wird ein Socket geöffnet, der unter dem File Descriptor 3 zur Verfügung steht. Dieser File Descriptor wird dann mit connect mit dem Endpunkt 134.76.12.3 auf dem TCP-Port 21 verbunden. Danach wird der File Descriptor mit dup dupliziert und kann ab sofort zusätzlich mit 5 angesprochen werden.

```
[...]  
read(3, "220-Welcome to ftp.gwdg.de\r\n220-...", 4096) = 40  
write(1, "220-Welcome to ftp.gwdg.de\n", 27) = 27  
[...]  
write(1, "Name (ftp.gwdg.de:daniel): ", 27) = 27  
read(0, "ftp\n", 1024) = 4  
write(5, "USER ftp\r\n", 10) = 10  
[...]
```

Listing 25.23 Fortsetzung von »/tmp/ftp.trace«

Die vom Socket (FD 3) mit `read` gelesenen Daten werden mit `write` auf die Standardausgabe (FD 1), also unser Terminal, geschrieben. Mit `read` wird dann der Benutzername, in diesem Fall `ftp`, von der Standardeingabe (FD 0) eingelesen und dann per `write` an den duplizierten Socket (FD 5) gesendet.

strace ist zweifelsohne ein mächtiges Werkzeug. Man darf sich aber nicht der Illusion hingeben, damit alle Probleme erkennen und beseitigen zu können. Es ist aber hervorragend geeignet, um Startprobleme von Programmen oder das Sterben von Prozessen zu analysieren, insbesondere dann, wenn keine brauchbaren Log-Meldungen ausgegeben werden.

Mithilfe von *strace* stellt man dann oft fest, dass einfach die Zugriffsrechte nicht korrekt gesetzt sind, Dateien fehlen, die Konfigurationsdatei doch anders heißt, als in der Manpage beschrieben wurde, oder dass das Programm einfach an den falschen Stellen nach Bibliotheken sucht.

25.1.8 Prüfung der Erreichbarkeit mit *my traceroute*

Das Programm *mtr* (*my traceroute*) ist ein wahrer Segen. Damit ist es möglich, nicht nur die Erreichbarkeit von Systemen, sondern gleichzeitig auch den Routenverlauf zu verfolgen.

Dieses Programm sollte zum Standardrepertoire gehören, da es so ungemein praktisch ist. Leider ist es standardmäßig nicht installiert, aber dafür in den Paketquellen enthalten.

Das Programm erwartet als Parameter analog zu *traceroute* einen Namen oder eine IP-Adresse. Rufen Sie es zum Beispiel mit `mtr 8.8.8.8` auf, erhalten Sie die Ausgabe aus Listing 25.24:

```
My traceroute [v0.95]  
example (192.168.0.47) -> 8.8.8.8 (8.8.8.8) 2023-01-03T15:17:02+0000  
Keys: Help Display mode Restart statistics Order of fields quit  
          Packets          Pings  
Host      Loss%  Snt  Last  Avg  Best  Wrst StDev  
1. 192.168.0.1      0.0%   4   0.4   0.4   0.4   0.5   0.0  
2. 10.131.64.1      0.0%   4   9.6   7.4   6.5   9.6   1.4  
3. 1413g-mx960-01-ae10-1310.dui.uni 0.0%   4   8.4   8.5   6.5  10.9   1.7
```

4.	de-fra01b-rc1.fra.unity-media.ne	0.0%	4	12.9	13.7	12.8	14.7	0.6
5.	1411g-mx960-01-ae3.nss.unity-med	0.0%	3	19.4	19.6	19.3	19.9	0.0
6.	google.dus.ecix.net	0.0%	3	19.0	20.2	19.0	22.1	1.6
7.	209.85.253.244	0.0%	3	19.2	19.4	18.8	20.1	0.0
8.	72.14.236.19	0.0%	3	20.1	19.9	19.4	20.2	0.0
9.	209.85.254.114	0.0%	3	20.6	23.6	19.2	30.9	6.4
10.	dns.google	0.0%	3	18.9	19.1	18.9	19.5	0.0

Listing 25.24 Ausgabe von »mtr«

Die Ausgabe aus Listing 25.24 wird pro Durchlauf aktualisiert. Neben dem Routenverlauf werden pro Routing-Hop die dazugehörigen Statistiken angezeigt. Verändert sich der Routenverlauf, wird dies durch Fettschrift gekennzeichnet.

Mit *mtr* haben Sie alle notwendigen Prüfungen im Blick: sowohl die Erreichbarkeit als auch das Routing. Ein wunderbares Stück Software!

25.1.9 Subnetzberechnung mit *ipcalc*

Die Berechnung von Subnetzen und Netzwerkmasken stellt auch einen geübten Administrator schon mal vor eine große Herausforderung. Aber Hilfe ist nicht weit, sie erfolgt in Form von *ipcalc*. Dieses kleine findige Tool nimmt Ihnen die komplexe Berechnung ab. Das Programm ist bei (fast) allen Distributionen Bestandteil der Paketquellen und kann daher über die gewohnten Mechanismen installiert werden.

Nach der Installation ist *ipcalc* direkt einsatzbereit. Zum Aufruf erwartet es nur die Angabe eines Netzes in CIDR-Schreibweise mit ausgeschriebener Netzwerkmaske oder einen Adressbereich. In Listing 25.25 haben wir für Sie die Ausgabe von *ipcalc* anhand des Netzes 172.31.253.0/21 dargestellt:

```
daniel@example:~$ ipcalc 172.31.253.0/21
Address: 172.31.253.0          10101100.00011111.11111 101.00000000
Netmask: 255.255.248.0 = 21  11111111.11111111.11111 000.00000000
Wildcard: 0.0.7.255          00000000.00000000.00000 111.11111111
=>
Network: 172.31.248.0/21     10101100.00011111.11111 000.00000000
HostMin: 172.31.248.1      10101100.00011111.11111 000.00000001
HostMax: 172.31.255.254    10101100.00011111.11111 111.11111110
Broadcast: 172.31.255.255  10101100.00011111.11111 111.11111111
Hosts/Net: 2046             Class B, Private Internet
```

Listing 25.25 Ausgabe von »ipcalc«

Aus dem übergebenen Netzwerk rechnet *ipcalc* alle notwendigen Informationen aus. Im oberen Teil (vor der Trennung durch =>) werden Informationen zur Netzwerkmaske ausgegeben.

Im unteren Teil befinden sich weitere Informationen, wie die Netzwerkadresse (*Network*), die erste und letzte Clientadresse (*HostMin* und *HostMax*), die Broadcast-Adresse und die maximale Anzahl an Clients im Netz (*Host/Net*). Dabei agiert *ipcalc* äußerst intelligent. Auch wenn Sie nicht die Netzwerkadresse (also die erste gültige Adresse im Netzwerk) angeben oder Ihnen die Netzwerkmaske nicht im CIDR-Format, sondern im Dezimalpunkt-Format vorliegt, gibt Ihnen *ipcalc* eine vernünftige Ausgabe zurück.

In Listing 25.26 werden *ipcalc* eine IP-Adresse und, durch ein Leerzeichen getrennt, eine Subnetzmaske im Dezimalpunkt-Format übergeben. Dies bringt *ipcalc* nicht aus dem Tritt; es liefert Ihnen trotzdem die korrekten Werte zurück:

```
daniel@example:~$ ipcalc 192.168.100.17 255.255.255.248
Address: 192.168.100.17      11000000.10101000.01100100.00010 001
Netmask: 255.255.255.248 = 29 11111111.11111111.11111111.11111 000
Wildcard: 0.0.0.7          00000000.00000000.00000000.00000 111
=>
Network: 192.168.100.16/29   11000000.10101000.01100100.00010 000
HostMin: 192.168.100.17     11000000.10101000.01100100.00010 001
HostMax: 192.168.100.22     11000000.10101000.01100100.00010 110
Broadcast: 192.168.100.23   11000000.10101000.01100100.00010 111
Hosts/Net: 6                 Class C, Private Internet
```

Listing 25.26 Die Vielfältigkeit und Intelligenz von »ipcalc«

Falls Sie nicht wissen, wie Sie einen Netzbereich am besten in Subnetze verpacken können, hilft Ihnen der Parameter *-r* dabei:

```
daniel@example:~$ ipcalc 172.31.0.0 172.31.17.0 -r ### openSUSE/CentOS: -d
deaggregate 172.31.0.0 - 172.31.17.0
172.31.0.0/20
172.31.16.0/24
172.31.17.0/32
```

Listing 25.27 Berechnung von optimalen Subnetzen: »ipcalc -r <START> <END>«

Wie Sie Listing 25.27 entnehmen können, berechnet *ipcalc* anhand der übergebenen Netze die optimale Verteilung der Subnetze – dies kann ein wahrer Segen sein.

25.2 Aus der Ferne – Remote-Administrationstools

In diesem Abschnitt wollen wir Ihnen ein paar kleine Programme vorstellen, die Ihnen aus der Ferne helfen können. Manchmal ist man nicht an seinem System, muss aber dennoch schnell auf einem Server nach dem Rechten sehen oder ein paar Dateien kopieren. Wenn das fremde System als Betriebssystem Windows einsetzt, steht man erst einmal vor einer Hürde.

Die *cmd* kennt leider kein *ssh* oder *scp*. Selbstverständlich gibt es Tools, die es auch Windows-Nutzern erlauben, sich mit Linux-Systemen zu verbinden und Dateien auszutauschen.

Wer kennt ihn nicht: den überladenen Schreibtisch? Doch auch hier gibt es softwaretechnische Abhilfe. So können Sie über eine Tastatur und Maus auch mehrere Rechner steuern. Dabei müssen Sie auf Vorzüge wie die Zwischenablage nicht verzichten. Für die Reisenden unter uns gibt es auch etwas Feines. Wer mit instabilen Internetverbindungen zu kämpfen hat (wie zum Beispiel im Zug), kennt das Phänomen der abreißenden SSH-Verbindungen. Alles steht, nichts geht mehr, und der Administrator schaut in die Röhre. Für all diese Probleme wollen wir Ihnen nun Lösungen vorstellen.

25.2.1 PuTTY

Eines der wohl wichtigsten Programme für den unter Microsoft Windows arbeitenden Linux- oder UNIX-Administrator ist unbestrittenermaßen *PuTTY* von Simon Tatham. Neben einem freien Telnet- und SSH-Client für Windows bietet die Programmsammlung rund um PuTTY auch einige andere Programme, die Ihnen das Leben erleichtern. Sie kann unter www.chiark.greenend.org.uk/~sgtatham/putty heruntergeladen werden. PuTTY ist weltweit millionenfach im Einsatz und gilt als stabile Software.

Nach dem Starten bekommen Sie die Dialogbox aus Abbildung 25.1 zu sehen. Tragen Sie einen Servernamen oder eine IP-Adresse ein, und verbinden Sie sich durch Klick auf OPEN.

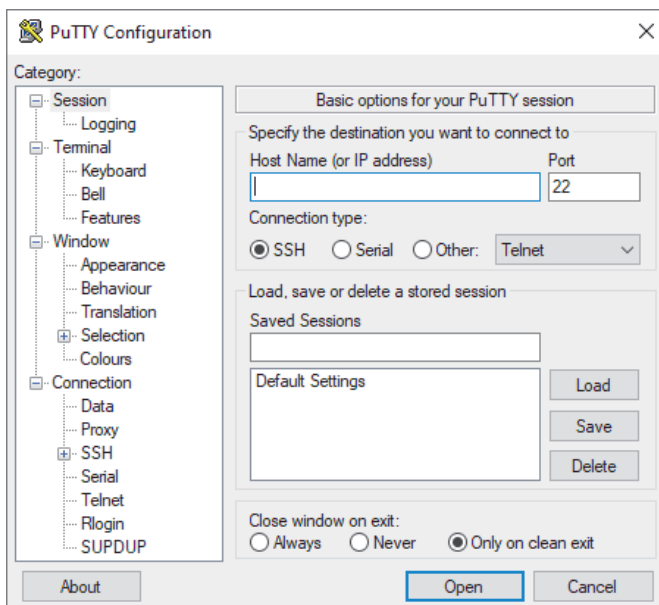


Abbildung 25.1 Der Hauptbildschirm von »PuTTY«

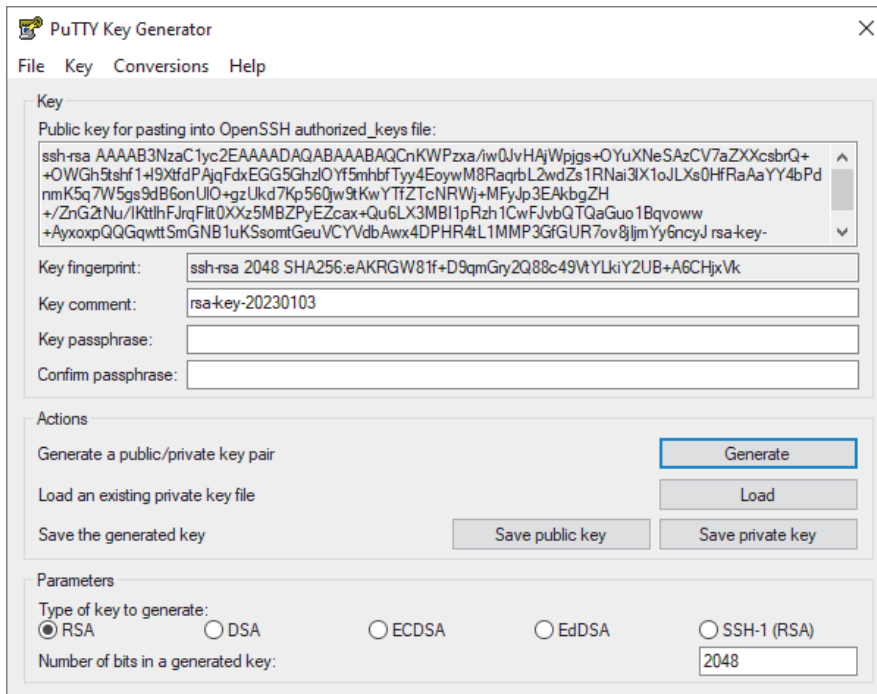
Das Programm stellt nun die Verbindung zum entfernten Server via SSH her und gibt Ihnen eine Konsolenverbindung. Im PuTTY-Fenster können Sie etwas mit der linken Maustaste markieren, es wandert dann ohne Umwege direkt ins Klemmbrett (Clipboard); und mit der rechten Maustaste können Sie Inhalte aus dem Klemmbrett einfügen.

Ein Klick mit der linken Maustaste auf das Feld links neben dem Titel des Fensters öffnet ein Menü, in dem Sie die folgenden Menüeinträge finden:

- ▶ **Special Command**
Hier können Sie Signale an Prozesse senden oder SSH-Schlüssel neu aushandeln lassen.
- ▶ **Event Log**
Informationen zum Verbindungsaufbau
- ▶ **New Session...**
zeigt eine Dialogbox, die es Ihnen erlaubt, eine neue Verbindung aufzubauen.
- ▶ **Duplicate Session**
öffnet ein neues Fenster mit einer neuen Verbindung zu dem Server, auf dem Sie in diesem Fenster arbeiten.
- ▶ **Saved Sessions**
bietet Ihnen eine Zugriffsmöglichkeit auf gespeicherte Sitzungen.
- ▶ **Change Settings...**
öffnet eine Dialogbox, in der Sie Einstellungen der aktuellen Session verändern können.
- ▶ **Copy All to Clipboard**
kopiert den kompletten Fensterinhalt, nicht nur den sichtbaren Bereich, ins Klemmbrett.
- ▶ **Clear Scrollback**
löscht den Scrollback-Buffer.
- ▶ **Reset Terminal**
setzt das Terminal zurück.
- ▶ **Full Screen**
vergrößert das Fenster bildschirmfüllend.
- ▶ **Help**
bietet Ihnen Zugriff auf die sehr gute Hilfe von PuTTY.
- ▶ **About PuTTY**
zeigt die Copyright-Notizen.

Mit PuTTY ist es ebenfalls – wie auch beim SSH-Client – möglich, sich per Schlüssel zu authentifizieren. Den Schlüssel generiert der *PuTTY Key Generator* (siehe Abbildung 25.2). Dazu klicken Sie im Programm auf **GENERATE** und bewegen die Maus, bis das Programm den Schlüssel erstellt hat. Speichern Sie – nach Eingabe eines Passworts (*Key passphrase*) – den

privaten und den öffentlichen Schlüssel ab. Im Programm wird ein öffentlicher Schlüssel angezeigt, den Sie genau so in die Datei `~/ssh/authorized_keys` auf Ihrem Server speichern.



25

Abbildung 25.2 Schlüsselgenerierung: »puttygen.exe«

Wenn Sie nun das Programm `pageant.exe` starten, erscheint ein Symbol in der Taskleiste Ihres Computers. Dort klicken Sie mit der rechten Maustaste und fügen den privaten Teil des Schlüssels hinzu, den Sie gerade erzeugt haben. Nach Eingabe des Passworts steht der Schlüssel nun PuTTY zur Verfügung. Um den Schlüssel in Zukunft schon beim Start von Windows verfügbar zu haben, fügen Sie eine Verknüpfung in Ihrem Autostart-Ordner nach folgendem Muster hinzu (passen Sie die Pfade an Ihre Installation an):

```
C:\Programme (x86)\putty\PAGEANT.EXE" H:\Personal\dvs.ppk
```

Listing 25.28 Laden des Schlüssels beim Starten

In der Programmsammlung um PuTTY sind noch drei wertvolle Kommandozeilenprogramme enthalten:

- ▶ Das erste ist *PuTTY Link* (`plink.exe`), mit dem Sie mit der Kommandozeile unter Windows Befehle an ein Linux- oder UNIX-System senden können.
- ▶ *PuTTY Secure File Transfer (SFTP) client* (`psftp.exe`) ist das Pendant zu `sftp` unter Linux.
- ▶ *PuTTY Secure Copy client* (`pscp.exe`) entspricht dem Programm `scp` unter Linux.

25.2.2 WinSCP

WinSCP ist ein Programm zur Übertragung von Daten zwischen Windows und anderen Systemen. Unterstützt werden die Protokolle SCP, SFTP, FTP und FTPS (siehe Abbildung 25.3).

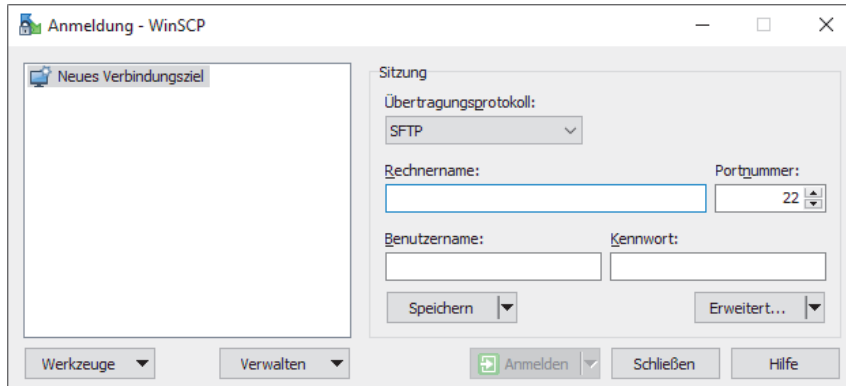


Abbildung 25.3 Login von »WinSCP«

Das von Martin Prikryl entwickelte Tool steht unter <http://winscp.net> zum Download bereit. Bei den *ssh*-Protokollen SCP und SFTP werden die gleichen Authentifizierungsmethoden wie auch bei PuTTY unterstützt. WinSCP liefert eine Kopie von Pageant und den PuTTY Key Generator (*puttygen*) mit aus. Im TOOLS-Menü gibt es die Möglichkeit, PuTTY-Session-Daten zu importieren. Vor der Anmeldung kann man im PREFERENCES-Menü wählen, ob man eine dem Norton Commander ähnliche Oberfläche haben möchte oder eine, die dem Explorer von Windows entspricht. Abbildung 25.4 zeigt die Norton-Commander-ähnliche Oberfläche.

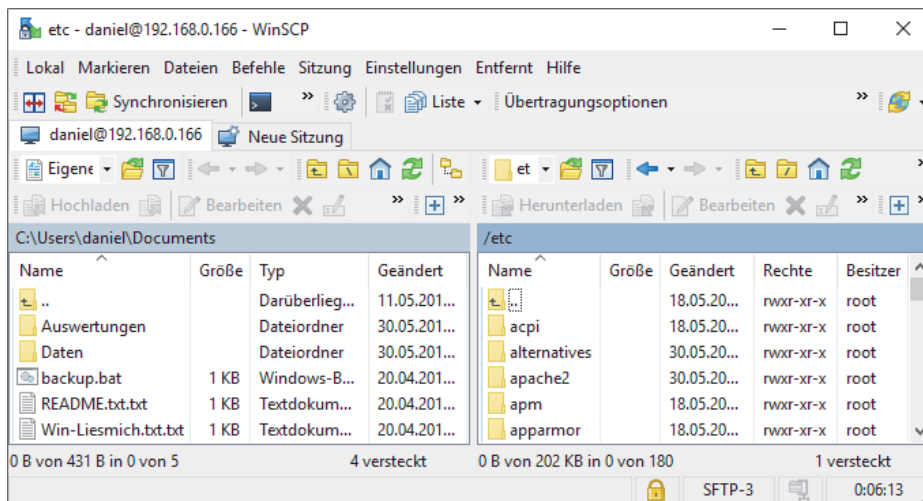


Abbildung 25.4 »WinSCP«-Commander-Ansicht

Eine Besonderheit von WinSCP ist, dass es sich im SENDEN AN-Menü des Explorers einnistet und dort – nach Rechtsklick auf eine Datei oder ein Verzeichnis – die Möglichkeit SENDEN AN WINSCP (FOR UPLOAD) bietet. Auf diese Weise ist es möglich, schnell Daten an ein System zu senden.

25.2.3 Synergy

Systemadministratoren haben auf ihren Schreibtischen meist mehr als ein Desktop-System. Dabei benötigen die doppelte und dreifache Ausführung der Tastatur und Maus natürlich viel Platz. Damit der Schreibtisch seine eigentliche Funktion erfüllen kann, nämlich die, darauf auch etwas schreiben zu können, empfiehlt sich der Einsatz von *Synergy*. Das Open-Source-Programm ist über GitHub unter <https://github.com/symless/synergy-core> erreichbar.

Synergy ermöglicht es, mehrere Systeme mit einer Tastatur und Maus zu steuern. Dabei arbeitet Synergy im Client/Server-Modus, und das betriebssystemübergreifend. Die Software steht sowohl für Windows als auch für Linux und macOS zur Verfügung (gegebenenfalls ist eine Lizenz erforderlich). Synergy ist in vielen Distributionen über das Paketmanagement verfügbar. Im Beispiel-Setup wird die Situation aus Abbildung 25.5 vorausgesetzt.

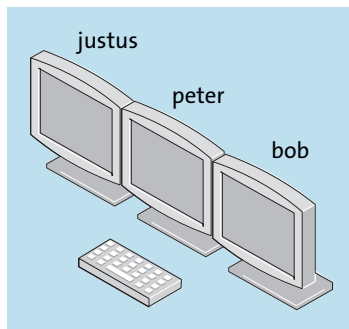


Abbildung 25.5 Beispiel-Setup »synergy«

Das System *peter* stellt die Tastatur und die Maus zur Verfügung und agiert somit als Server. Die Systeme *justus* und *bob* hingegen agieren als Clients, die von *peter* gesteuert werden sollen. Installieren Sie auf allen Systemen *Synergy*.

Auf dem Server (*peter*) müssen Sie die Konfiguration des Setups vornehmen. Editieren Sie dafür die Datei `/etc/synergy.conf`:

```
section: screens
    justus:
    bob:
    peter:
end
```

```
section: links
    justus:
        right = peter
    peter:
        left = justus
        right = bob
    bob:
        left = peter
end
```

Listing 25.29 »/etc/synergy.conf«

Synergy verfolgt bei der Angabe von Positionen das Richtungsprinzip. Das heißt, die Konfigurationsdatei ist in Sektionen unterteilt. In der Sektion `screens` werden die Systeme definiert, dabei werden die Hostnamen verwendet. In der Sektion `links` wird die Position der Systeme zueinander bestimmt. Im Beispiel hat *peter* die Tastatur und die Maus.

Sobald die Maus links aus dem Bild fährt, aktiviert Synergy die Umleitung der Eingabe auf das System *justus*. Fährt die Maus von *justus* nach rechts aus dem Bild, wird die Eingabe wieder auf *peter* geleitet.

Äquivalent geht es von *peter* nach rechts zu *bob* und zurück. Sobald der Server mit `synergys` gestartet wurde, können die Clients sich verbinden. Der Synergy-Server öffnet den TCP-Port 24800 und wartet auf eingehende Verbindungen.

Da in der Konfiguration auf dem Server bereits alle Relationen der Bildschirme zueinander festgelegt wurden, müssen die Clients nicht separat konfiguriert werden. Sie müssen lediglich den Server angeben:

```
user@bob:~# synergyc peter.example.net
```

Listing 25.30 Synergy-Client starten: »bob«



Unterschiedliche Programme: Client und Server

Achten Sie darauf, auf dem Client das Programm `synergyc` und auf dem Server das Programm `synergys` zu starten, da ansonsten keine Verbindung hergestellt werden kann.

Die Konfiguration eines Windows-Clients ist ähnlich simpel. Nach der Installation startet Synergy mit dem Fenster, in dem Sie nur den Namen des Servers angeben müssen.

Zwischen den Systemen wird zusätzlich zu Tastatur und Maus auch die Zwischenablage übergeben. So können Sie Texte, die Sie unter Windows mit `Strg+C` kopiert haben, unter Linux-Systemen mit der mittleren Maustaste einfügen. Selbstverständlich funktioniert das auch umgekehrt.

25.2.4 Eine für immer: mosh

Das Programm *mosh* stellt eine *mobile Shell* zur Verfügung. Es verfolgt dabei einen anderen Ansatz als zum Beispiel *screen*, bei dem eine Shell auf dem Server gestartet wird, mit der Sie sich wieder verbinden können. *mosh* bietet die gleiche Funktionalität wie SSH, hat aber nicht die gleichen Schwächen. Sie können zwischen unterschiedlichen Medien wechseln, ohne dass die Verbindung getrennt wird. Wird die Verbindung einmal getrennt, nimmt *mosh* den Dienst direkt wieder auf, sobald eine Verbindung wieder aufgebaut werden kann. Dabei setzt *mosh* auf UPD (statt auf TCP wie bei SSH) und verwendet *diff* und *patch*, um Bildschirm-inhalte abzugleichen.

Hier sind die Vorteile von *mosh* im Überblick:

- ▶ **Wechsel der Internetverbindung**
Sie können das Medium (LTE, WLAN, LAN etc.) ohne Verbindungsverlust beliebig oft wechseln.
- ▶ **Wiederaufnahme**
Die Verbindung wird stets wieder aufgenommen – zum Beispiel nach Aktivierung des Ruhemodus.
- ▶ **Niedriger Bandbreitenbedarf**
mosh ist auch bei schlechten Leitungen benutzbar.
- ▶ **Statusanzeige**
Falls ein Ereignis eintritt – zum Beispiel der Verbindungsabbruch –, wird Ihnen eine entsprechende Meldung angezeigt (oberste Zeile). Der Benutzer wird nie ohne einen Hinweis auf den Grund des Abbruchs zurückgelassen.
- ▶ **Setzt auf SSH auf**
Daher sind keine zusätzlichen Portfreigaben notwendig.

mosh ist Bestandteil aller gängigen Distributionen und kann somit über die Paketquellen installiert werden. Über die Projektseite <https://mosh.org> können Sie stets die aktuellen Neuigkeiten und Pakete beziehen.

mosh setzt auf SSH auf und arbeitet daher auch im Client/Server-Modus. Dabei sind aber Client und Server identisch – *mosh* identifiziert selbstständig, wer Client und wer Server ist.

Nach der Installation auf beiden Seiten (auf der Serverseite wird ein lauffähiger *openssh*-Server vorausgesetzt) ist *mosh* direkt einsatzbereit. Verbinden Sie sich einfach auf Ihren Server mittels `mosh <USERNAME>@<SERVER>` – analog zu SSH können Sie diesen bei gleichem Benutzernamen auch weglassen.

Anschließend verhält *mosh* sich wie SSH – nur mit den zusätzlichen Funktionen. In Abbildung 25.6 haben wir für Sie die Ausgabe dargestellt, die erscheint, wenn *mosh* seine Verbindung einmal verliert.

```

mosh: Last contact 11 seconds ago. [To quit: Ctrl-^ .]
daniel@saturn:/etc$ cd bind/
daniel@saturn:/etc/bind$ ls -l
insgesamt 52
-rw-r--r-- 1 root root 2389 Jan 13 21:54 bind.keys
-rw-r--r-- 1 root root 237 Jan 13 21:54 db.0
-rw-r--r-- 1 root root 271 Jan 13 21:54 db.127
-rw-r--r-- 1 root root 237 Jan 13 21:54 db.255
-rw-r--r-- 1 root root 353 Jan 13 21:54 db.empty
-rw-r--r-- 1 root root 270 Jan 13 21:54 db.local
-rw-r--r-- 1 root root 3048 Jan 13 21:54 db.root
-rw-r--r-- 1 root bind 463 Jan 13 21:54 named.conf
-rw-r--r-- 1 root bind 490 Jan 13 21:54 named.conf.default-zones
-rw-r--r-- 1 root bind 165 Jan 13 21:54 named.conf.local
-rw-r--r-- 1 root bind 890 Jan 25 11:56 named.conf.options
-rw-r----- 1 bind bind 77 Jan 25 11:56 rndc.key
-rw-r--r-- 1 root root 1317 Jan 13 21:54 zones.rfc1918
daniel@saturn:/etc/bind$ █

```

Abbildung 25.6 Anzeige von Verbindungsverlust

Falls Sie versuchen sollten, mittels *mosh* auf ein System zuzugreifen, auf dem keine Installation vorhanden ist, wird dies so wie in Listing 25.31 quittiert:

```

daniel@saturn:~$ mosh merkur
daniel@merkur's password:
bash: mosh-server: Befehl nicht gefunden
Connection to merkur closed.
/usr/bin/mosh: Did not find mosh server startup message. \
(Have you installed mosh on your server?)

```

Listing 25.31 Fehlerausgabe, wenn »mosh« auf der Gegenseite nicht installiert ist

Kapitel 26

Versionskontrolle

»Zurück in die Zukunft«

26

Versionskontrollsysteme oder auch Versionsverwaltungssysteme, im Englischen *Version Control Systems* (VCS), bieten die Möglichkeit, »in die Vergangenheit zu reisen« und alte Stände von Dateien wiederherzustellen sowie verschiedene Versionen von Dateien miteinander zu vergleichen. Versionskontrollsysteme werden überwiegend in der Softwareentwicklung zur Verwaltung von Quelltexten eingesetzt, leisten aber auch in der Systemadministration zur Verwaltung von Konfigurationsdateien gute Dienste.

In diesem Kapitel werden Versionskontrollsysteme nur als Mittel zur Verwaltung von Konfigurationsdateien behandelt. Insbesondere wird davon ausgegangen, dass nur wenige Personen an den Dateien arbeiten. Das ganze Aufgabenfeld rund um Konflikte beim Zusammenführen (*Mergen*) von unterschiedlichen Versionsständen wird dabei bewusst ausgespart. Die Systeme erfüllen dabei folgende Vorgaben:



- ▶ **Archivierung**
von alten Ständen, zum Nachvollziehen, wie sich eine Konfiguration entwickelt hat, und um alte Stände bei Bedarf wiederherstellen zu können
- ▶ **Protokollierung**
Es kann jederzeit geprüft werden, wer zu welcher Zeit was verändert hat.
- ▶ **Wiederverwendbarkeit**
Auch wenn eine alte Konfiguration auf einem neuen System eventuell nicht mehr das tut, was sie tun soll, so kann sie dennoch als Vorgabe für Konfigurationen auf Altsystemen dienen. Das ist ein Spezialfall der Archivierung.

Um diese Vorgaben zu erfüllen, haben sich in den letzten Jahren verschiedene Sichtweisen herausgebildet, und innerhalb der verschiedenen Sichtweisen existieren unterschiedliche Implementationen dieser Sichtweisen. Im Großen und Ganzen lassen sich drei verschiedene Philosophien – *lokal*, *zentral* und *dezentral* – unterscheiden, die in Abschnitt 26.1, »Philosophien«, näher beschrieben werden. Es gibt eine ganze Reihe von Versionskontrollsystemen, die als Open-Source-Software veröffentlicht wurden, und es existieren ebenfalls einige Vertreter aus dem Lager der Closed-Source-Software. In Tabelle 26.1 finden Sie einige Beispiele für die verschiedenen Versionskontrollsysteme, die der Wikipedia¹ entnommen wurden.

¹ <https://de.wikipedia.org/wiki/Versionsverwaltung>

	Open-Source-Systeme	Proprietäre Systeme
Zentrale Systeme	SCCS RCS CVS Subversion (SVN)	Alienbrain Perforce Team Foundation Server Visual SourceSafe ClearCase IBM Ration Synergy PTC Integrity SAP Design in Time Repository (DTR) versiondog Sourcegear Vault
Verteilte Systeme	Bazaar Bitkeeper Darcs Fossil Git GNU arch Mercurial Monotone	Rational Team Concert

Tabelle 26.1 Beispiele für Versionskontrollsysteme aus der Wikipedia

26.1 Philosophien

Es gibt viele verschiedene Ansätze für die Versionskontrolle, aber im Großen und Ganzen lassen sich drei verschiedene Philosophien unterscheiden, die wir im Folgenden betrachten.

26.1.1 Lokal

Alle Versionen von Dateien werden lokal gespeichert und nicht an einen Server übertragen. Als einfachstes Beispiel mag Listing 26.1 dienen:

```
$ cp config config.alt
$ edit config
```

Listing 26.1 Einfachste Art der lokalen Versionskontrolle

Die originale Datei *config* wird damit in *config.alt* kopiert, und die ursprüngliche Datei wird verändert. Nun existieren zwei Versionen der Datei. Es gibt Systeme und sogar Dateisysteme, die ein solches Vorgehen unterstützen und die Dateien rotierend benennen. Aus *config* wird *config.1*, bei der nächsten Änderung wird aus *config.1* die Datei *config.2*, und aus *config* wird erneut *config.1*, so wie es Listing 26.2 zeigt:


```
$ cp config.1 config.2
$ cp config config.1
$ edit config
```

Listing 26.2 Lokale Versionskontrolle mit verschiedenen Versionen

Das lässt sich beliebig weiterdenken oder durch einen Schwellenwert – beispielsweise zwei Versionen wie in Listing 26.3 – abbrechen:

```
$ rm config.2
$ cp config.1 config.2
$ cp config config.1
$ edit config
```

Listing 26.3 Lokale Versionskontrolle, rotierend

Wenn die *config* ein weiteres Mal verändert wird, dann wird die älteste Version gelöscht. Allgemein üblich ist es, statt der Nummern Daten bzw. Zeitstempel einzusetzen. Damit ist auf einen Blick ersichtlich, bis zu welchem Zeitpunkt eine Konfigurationsdatei gültig war, wie Listing 26.4 zeigt:

```
$ cp config config.20230530
$ edit config
```

Listing 26.4 Lokale Versionskontrolle mit Zeitstempel

26.1.2 Zentral

Einen Schritt weiter gehen zentrale Versionskontrollsysteme. Dort wird mit den Begriffen *Arbeitskopie* und *Repository* unterschieden, in welchem Teil der Arbeitskette man sich befindet. Dateien werden in der Arbeitskopie verändert und nach Abschluss der Arbeiten – zusammen mit einer Nachricht – an eine zentrale Stelle, einen zentralen Server, geschickt. Der Server, der das Repository verwaltet, besitzt alle Versionsstände der verwalteten Dateien. Lokal in der Arbeitskopie liegt immer nur eine einzige Version der Datei vor. Will man sich Unterschiede zwischen verschiedenen Versionen anzeigen lassen, so wird für jede Version der Server beauftragt, die angeforderte Version zu schicken.

Der große Vorteil eines solchen Verfahrens besteht darin, dass lokal eine sehr aufgeräumte Arbeitskopie vorliegt, ohne dass man Dateien in vielen verschiedenen Versionen vorliegen hat, und dass zentral auf dem Server ein Backup aller alten (und auch der aktuellen) Konfigurationen vorliegt. In einer Pseudosprache sieht der Arbeitsablauf so wie in Listing 26.5 beschrieben aus:

```
$ vcs get datei
** Version 42 von datei ausgecheckt **
$ edit datei
```

```
$ vcs send datei "Parameter x auf y gesetzt"  
** Version 43 von datei eingecheckt, Meldung "Parameter x auf y gesetzt" **
```

Listing 26.5 Arbeitsablauf bei einem zentralen Versionskontrollsystem

Vor der Bearbeitung wird die aktuelle Version einer Datei geholt. Diese wird bearbeitet, und die bearbeitete Datei wird wieder zurück an das zentrale System gesendet. Die Versionsnummern werden von der zentralen Stelle verwaltet.

26.1.3 Dezentral

Einen der größten Nachteile von zentralen Versionskontrollsystemen, nämlich die Tatsache, dass der zentrale Server immer verfügbar sein muss, beheben dezentrale Versionskontrollsysteme. Sie gelten auch als die zeitgemäße Variante der Versionskontrolle. Jedes ausgecheckte Repository enthält die komplette Historie aller jemals durchgeführten Änderungen in komprimierter Form. Aus diesem Grund spricht man bei der Kopie auch von *Klonen*. Unterschiede zwischen Versionen oder zwischen der aktuellen Arbeitskopie und historischen Ständen lassen sich anzeigen, ohne dass eine Verbindung zu einem zentralen Server nötig wäre. Dieser Vorteil wird allerdings durch einen zusätzlichen Schritt erkauft. Änderungen, die bei zentralen Systemen an einen Server übertragen werden, müssen erst lokal festgeschrieben werden, bevor der aktuelle Stand an den Server übertragen wird:

```
$ vcs fetch  
** Synchronisierung von Aenderungen **  
$ edit datei  
$ vcs send datei "Parameter x auf y gesetzt"  
** datei wurde lokal festgeschrieben **  
$ vcs transfer  
** Alle lokalen Aenderungen an Server uebertragen **
```

Listing 26.6 Arbeitsablauf bei einem dezentralen Versionskontrollsystem

Zunächst beginnt in Listing 26.6 der Arbeitsablauf mit einer Synchronisation des eigenen Repositories mit dem entfernten. Das ist nicht zwingend nötig, da alle Änderungen auch lokal vorhanden sind, aber es erleichtert das spätere Zusammenführen von Klonen erheblich. Anschließend wird die Datei bearbeitet und lokal festgeschrieben (wobei die Änderung mit einer Meldung versehen wird). In einem letzten Schritt werden alle Änderungen an einen anderen Server oder an ein anderes Repository übertragen.

26.2 Versionskontrollsysteme

Versionskontrollsysteme gibt es wie Sand am Meer. Die englischsprachige Wikipedia-Seite https://en.wikipedia.org/wiki/Comparison_of_version_control_software listet aktuell bereits

35 verschiedene Systeme und zeigt ihre verschiedenen Stärken und Schwächen auf. In diesem Abschnitt werden nur die fünf Systeme behandelt, die im Open-Source-Umfeld am häufigsten anzutreffen sind. Von den zentralen Systemen stellen wir *CVS* und *Apache Subversion* vor, wobei die Nutzung von CVS sehr stark zurückgegangen ist. CVS wird auch nicht mehr aktiv weiterentwickelt. Bei den dezentralen Systemen finden sich *GNU Bazaar*, *Mercurial* und *Git*, wobei Git die weiteste Verbreitung findet, gefolgt von Mercurial und zum Schluss GNU Bazaar. Mitte 2023 sah die Verteilung auf *openhub.net* (siehe <https://www.openhub.net/repositories/compare>, einer Seite, die Auswertungen vielfältiger Open-Source-Projekte durchführt) so aus, wie Tabelle 26.2 zeigt.

Versionskontrollsystem	Herbst 2018	Anfang 2021	Frühling 2023
Git	60 %	72 %	74 %
Apache Subversion	33 %	23 %	22 %
CVS	2 %	1 %	1 %
Mercurial	1 %	1 %	1 %
GNU Bazaar	1 %	0 %	0 %

Tabelle 26.2 Verbreitung von Versionskontrollsystemen

Die Daten aus Tabelle 26.2 sind nicht repräsentativ, da es sich um selbst gemeldete und öffentliche Repositories handelt, aber sie geben einen guten Anhaltspunkt über die Verbreitung.

26.2.1 CVS

CVS (*Concurrent Versioning System*, siehe <https://savannah.nongnu.org/projects/cvs>) begann 1989 als Weiterentwicklung von RCS, dem *Revision Control System*. CVS benötigt die Umgebungsvariable `CVSROOT`, um auf Repositories zugreifen zu können. Ein neues Repository erzeugt man mit `cvs init` (siehe Listing 26.7):

```
$ export CVSROOT=/var/tmp/cvsrepository
$ cvs init
```

Listing 26.7 Initialisierung eines Repositorys mit CVS

Danach kann man beliebige Verzeichnisse in dieses Repository importieren, beispielsweise */etc*. In Listing 26.8 wird die Benutzung des `import`-Befehls gezeigt:

```
$ cd /etc
$ cvs import -m "Initialer Import" etc Dirk start
N etc/netconfig
```

```
N etc/securetty
[...]
N etc/insserv.conf.d/rpcbind
```

No conflicts created by this import

Listing 26.8 Import eines Verzeichnisses bei CVS

Verallgemeinert folgt der `import`-Befehl folgendem Muster:

```
cvs import -m Log-Nachricht Modul Hersteller-Information Release-Information
```

Mit den Angaben aus Listing 26.8 legen wir fest, dass das Modul *etc* heißen soll, die Herstellerinformation ist *Dirk*, und die Release-Information lautet *start*. Der Aufbau von CVS-ROOT ist abhängig von der Art und Weise, wie man das Repository auschecken möchte: CVS-ROOT=<Methode:><Name:><Verzeichnis>.

Dabei können für *Methode* verschiedene Werte eingetragen werden. Das Feld muss leer sein, wenn ein lokal erreichbares Verzeichnis das Ziel ist. Alternativ kann es den Wert `:ext` für den Zugriff über Remote Shell oder SSH enthalten, oder es nimmt den Wert `:pserver` für den Zugriff über ein CVS-eigenes Protokoll an. *Name* setzt sich aus dem User- und dem Rechnernamen zusammen. `username@hostname` besagt, dass der CVS-Server auf dem Server *hostname* läuft und sich *username* mit diesem verbinden will. Wenn man lokal arbeitet, kann der *hostname* weggelassen werden. Wenn der aktuelle User sich verbinden soll, dann kann man den *username* ebenfalls weglassen. Zum Schluss gibt *Verzeichnis* das Hauptverzeichnis an, in dem die Repositories verwaltet werden. Das gerade angelegte Modul wird jetzt im Homeverzeichnis des angemeldeten Benutzers wieder ausgecheckt, wie in Listing 26.9 beschrieben:

```
$ export CVSROOT=/var/tmp/cvsrepository
$ cd $HOME
$ cvs checkout etc
cvs checkout: Updating etc/python2.7
cvs checkout: Updating etc/rc0.d
cvs checkout: Updating etc/rc1.d
[...]
U etc/w3m/mailcap
U etc/xml/catalog
U etc/xml/xml-core.xml
```

Listing 26.9 Checkout eines Moduls mit CVS

Mit den Dateien im ausgecheckten Modulverzeichnis kann jetzt gearbeitet werden (siehe Listing 26.10):

```
$ cd $HOME/etc
$ vim hosts
```

```

$ cvs diff hosts
Index: hosts
=====
RCS file: /var/tmp/cvsrepository/etc/hosts,v
retrieving revision 1.1.1.1
diff -r1.1.1.1 hosts
1c1
< 127.0.0.1    localhost
---
> 127.0.0.1    localhost meincomputer

$ cvs commit -m "meincomputer zu hosts hinzugefuegt"
cvs commit: Examining .
[...]
cvs commit: Examining xml
/var/tmp/cvsrepository/etc/hosts,v <-- hosts
new revision: 1.2; previous revision: 1.1

```

Listing 26.10 Arbeit mit dem ausgecheckten Verzeichnis bei CVS

In Listing 26.10 sehen Sie die einzelnen Schritte. Dem Wechsel in das Arbeitsverzeichnis folgt die Bearbeitung der Datei *hosts*. Der `cvs diff` zeigt die Unterschiede zwischen der lokalen Datei und dem Repository, schließlich wird mit `cvs commit` die Änderung an das Repository übertragen. Wie Sie feststellen können, sucht CVS im gesamten Verzeichnis nach Änderungen und würde diese alle auf einmal übertragen. Listing 26.11 zeigt alle Änderungen, die an der Datei *hosts* vorgenommen wurden:

```

$ cvs log hosts
RCS file: /var/tmp/cvsrepository/etc/hosts,v
Working file: hosts
head: 1.2
branch:
locks: strict
access list:
symbolic names:
    start: 1.1.1.1
    Dirk: 1.1.1
keyword substitution: kv
total revisions: 3;    selected revisions: 3
description:
-----
revision 1.2
date: 2014-07-27 19:06:22 +0200; author: dirk; state: Exp; lines: +1 -1; \
    commitid: 10053D5318E0BCD08F4;

```

```
meincomputer zu hosts hinzugefuegt
-----
revision 1.1
date: 2014-07-27 18:30:32 +0200; author: root; state: Exp; \
  commitid: 10053D529280B56DA03;
branches: 1.1.1;
Initial revision
-----
revision 1.1.1.1
date: 2014-07-27 18:30:32 +0200; author: root; state: Exp; lines: +0 -0; \
  commitid: 10053D529280B56DA03;
Initialer Import
=====
```

Listing 26.11 Log-Einträge anschauen mit CVS

Detailliertere Informationen finden sich im frei zugänglichen Buch »Open Source Development with CVS, 3rd Edition« von Karl Fogel und Moshe Bar unter <https://cvsbook.red-bean.com/>.

26.2.2 Apache Subversion

Die Entwicklung von SVN, *Apache Subversion* (<https://subversion.apache.org>), begann im Jahr 2000 bei CollabNet, und im Februar 2004 erschien die erste stabile Version. Apache Subversion ist damit wesentlich jünger als CVS.

```
$ svnadmin create /var/tmp/svnrepository
$ cd $HOME
$ svn checkout file:///var/tmp/svnrepository
Checked out revision 0.
```

Listing 26.12 Ein Repository mit Apache Subversion erzeugen

Listing 26.12 zeigt, dass Apache Subversion es uns Benutzern etwas leichter macht: Mit `svnadmin create` wird ein Repository angelegt, und mit `svn checkout` wird es ausgecheckt.

Wir bearbeiten in Listing 26.13 die Datei *liesmich* und schauen mit `svn status` nach, ob es im aktuellen Verzeichnis Änderungen gibt, die sich noch nicht im Repository befinden. Diese Änderungen fügen wir mittels `svn add` der Versionskontrolle hinzu und übertragen sie per `svn commit` an das Repository (siehe Listing 26.13):

```
$ svn status
?      liesmich
$ svn add liesmich
A      liesmich
$ svn commit -m "Lies das" liesmich
```

```
Adding      liesmich
Transmitting file data .
Committed revision 1.
```

Listing 26.13 Mit Apache Subversion Dateien unter Versionskontrolle stellen

In Listing 26.14 sehen Sie, wie man mit Apache-Subversion-Befehlen herausfinden kann, was sich an einer Datei verändert hat:

```
$ svn diff liesmich
Index: liesmich
=====
--- liesmich      (revision 1)
+++ liesmich      (working copy)
@@ -1,3 @@
    Lies das!
+
+Mehr habe ich nicht zu sagen.
```

```
$ svn commit -m "mehr nicht"
Sending      liesmich
Transmitting file data .
Committed revision 2.
```

Listing 26.14 Ändern einer Datei mit Apache Subversion

Die Datei *liesmich* wird in Listing 26.14 weiterbearbeitet, und mit `svn diff` finden wir den Unterschied zur Datei im Repository heraus. Diese Änderungen werden wie gewohnt an das Repository übertragen. Mit dem `svn log`-Kommando können wir uns in Listing 26.15 alle Änderungen an der Datei anzeigen lassen:

```
$ svn log liesmich
-----
r2 | root | 2014-07-27 19:21:14 +0200 (Sun, 27 Jul 2014) | 1 line
mehr nicht
-----
r1 | root | 2014-07-27 19:20:30 +0200 (Sun, 27 Jul 2014) | 1 line

Lies das
-----
```

Listing 26.15 Anzeigen aller Änderungen bei Apache Subversion

Weitere Informationen können Sie im deutschsprachigen und frei verfügbaren Buch »Versions-Kontrolle mit Subversion« von Ben Collins-Sussman, Brian W. Fitzpatrick und C. Michael Pilato unter <http://svnbook.red-bean.com> nachlesen.

26.2.3 GNU Bazaar

GNU Bazaar (kurz *BZR*, siehe <https://bazaar.canonical.com>) ist im März 2005 als Nachfolger des Systems *baz* entstanden und wurde maßgeblich vom britischen Unternehmen Canonical Ltd. (der Firma hinter Ubuntu Linux) unterstützt. Nachdem Canonical Anfang 2012 fast alle Entwickler abgezogen hat, ist die Entwicklung nahezu eingeschlafen.

Alle drei vorgestellten dezentralen Versionskontrollsysteme haben gemeinsam, dass sie anmahnen, die Userinformationen anzugeben, wenn man in ein Repository schreibt, und so beginnt die Konfiguration mit der einmaligen Angabe der Userinformationen:

```
$ bzz whoami "Dirk Deimeke <dirk@deimeke.net>"
$ cat ~/.bazaar/bazaar.conf
[DEFAULT]
email = Dirk Deimeke <dirk@deimeke.net>
```

Listing 26.16 Eingeben der Userinformationen bei GNU Bazaar

GNU Bazaar erlaubt das Anlegen eines Repositorys mit verschiedenen Teilbäumen. Im Beispiel aus Listing 26.17 wird das *bzz*-Repository angelegt und darin der Zweig *trunk* erstellt, der für die weiteren Schritte benutzt wird:

```
$ bzz init-repository /var/tmp/bzzrepository
Shared repository with trees (format: 2a)
Location:
  shared repository: /var/tmp/bzzrepository
$ bzz init /var/tmp/bzzrepository/trunk
Created a repository tree (format: 2a)
Using shared repository: /var/tmp/bzzrepository/
$ bzz branch /var/tmp/bzzrepository/trunk
Branched 0 revisions.
```

Listing 26.17 Erzeugen eines Repositorys mit GNU Bazaar

In Listing 26.18 wird eine *liesmich*-Datei angelegt. Das *status*-Kommando bezeichnet diese Datei als nicht unter Versionskontrolle stehend. Mit *add* wird sie als zugehörig markiert und mit *commit* lokal übertragen. Der *push* überträgt schließlich alle lokalen Änderungen an das »entfernte« Repository (siehe Listing 26.18):

```
$ vim liesmich
$ bzz status
unknown:
  liesmich
$ bzz add liesmich
adding liesmich
$ bzz commit -m "Lies das" liesmich
```



```

Committing to: /home/dirk/trunk/
added liesmich
Committed revision 1.
$ bzip push :parent
All changes applied successfully.
Pushed up to revision 1.

```

Listing 26.18 Mit GNU Bazaar Dateien unter Versionskontrolle stellen

In Listing 26.19 werden die Befehlsausgaben gezeigt, die erwartet werden können, wenn sich Dateien ändern. Neu ist hier der `diff`-Befehl, der die Unterschiede zwischen der lokalen Datei und der aktuellen Datei im Repository aufzeigt.

```

$ bzip status
modified:
  liesmich
$ bzip diff liesmich
=== modified file 'liesmich'
--- liesmich      2014-07-30 17:05:39 +0000
+++ liesmich      2014-07-30 17:06:05 +0000
@@ -1,1 +1,3 @@
  Lies das!
+
+Mehr habe ich nicht zu sagen.
$ bzip commit -m "mehr nicht" liesmich
Committing to: /home/dirk/trunk/
modified liesmich
Committed revision 2.
$ bzip push :parent
All changes applied successfully.
Pushed up to revision 2.

```

Listing 26.19 Ändern einer Datei mit GNU Bazaar

Zum guten Schluss zeigt der `log`-Befehl alle Änderungen einer Datei (siehe Listing 26.20):

```

$ bzip log liesmich
-----
revno: 2
committer: Dirk Deimeke <dirk@deimeke.net>
branch nick: trunk
timestamp: Wed 2014-07-30 19:06:31 +0200
message:
  mehr nicht
-----
revno: 1

```

```
committer: Dirk Deimeke <dirk@deimeke.net>
branch nick: trunk
timestamp: Wed 2014-07-30 19:05:39 +0200
message:
  Lies das
```

Listing 26.20 Anzeige aller Änderungen bei GNU Bazaar

Versionskontrollsysteme, und GNU Bazaar ist da keine Ausnahme, gehören mit zu den am besten dokumentierten Open-Source-Produkten. Daher ist der Userguide von GNU Bazaar eine echte Empfehlung: <http://doc.bazaar.canonical.com/bzr.dev/en/user-guide/index.html>

26.2.4 Mercurial

Mercurial (kurz *HG*, siehe <https://www.mercurial-scm.org/>) startete ebenfalls im Jahr 2005 und wurde initiiert, um einen Nachfolger des Systems *BitKeeper* zu schaffen, mit dem damals der Linux-Kernel entwickelt wurde und das durch eine Lizenzänderung nicht mehr frei verwendbar war. Heute ist Mercurial in der Community rund um die Programmiersprache *Python* weit verbreitet. Die Befehle bei Mercurial sehen nicht sehr unterschiedlich aus im Vergleich zu dem, was Sie bereits bei GNU Bazaar kennengelernt haben. Ein großer Unterschied ist, dass Mercurial weniger gesprächig ist und dass es deutlich weniger Ausgaben liefert. Mercurial nutzt das Kommando `hg`. Das leitet sich vom Kürzel für Quecksilber im Periodensystem der Elemente her: *Mercurial* ist das englische Wort für *Quecksilber* bzw. *quecksilberhaltig*. Anders als bei GNU Bazaar und Git müssen bei Mercurial die Userinformationen in eine Datei geschrieben werden (siehe Listing 26.21). Es gibt kein Kommando dafür.

```
[ui]
username = Dirk Deimeke <dirk@deimeke.net>
```

Listing 26.21 Userinformationen für Mercurial in `/.hgrc`

Das Erstellen eines Repositories geht direkt, wie Listing 26.22 zeigt:

```
$ hg init /var/tmp/hgrepository
$ hg clone /var/tmp/hgrepository
destination directory: hgrepository
updating to branch default
0 files updated, 0 files merged, 0 files removed, 0 files unresolved
```

Listing 26.22 Erzeugen eines Repositories bei Mercurial

Auch die Befehle, um eine Datei unter Versionskontrolle zu stellen, stimmen mit denen von GNU Bazaar überein (siehe Listing 26.23):

```

$ hg status
? liesmich
$ hg add liesmich
$ hg commit -m "Lies das" liesmich
$ hg push
pushing to /var/tmp/hgrepository
searching for changes
adding changesets
adding manifests
adding file changes
added 1 changesets with 1 changes to 1 files

```

Listing 26.23 Dateien unter Versionskontrolle stellen mit Mercurial

Auch bei geänderten Dateien finden sich in Listing 26.24 keine Überraschungen; einzig die Ausgaben sind andere:

```

$ vim liesmich
$ hg status
M liesmich
$ hg diff liesmich
diff -r f34871bf678a liesmich
--- a/liesmich Wed Jul 30 19:11:07 2014 +0200
+++ b/liesmich Wed Jul 30 19:13:12 2014 +0200
@@ -1,1 +1,3 @@
  Lies das!
+
+Mehr habe ich nicht zu sagen.
$ hg commit -m "mehr nicht" liesmich
$ hg push
pushing to /var/tmp/hgrepository
searching for changes
adding changesets
adding manifests
adding file changes
added 1 changesets with 1 changes to 1 files

```

Listing 26.24 Ändern einer Datei bei Mercurial

Auch bei der Ausgabe des log-Kommandos zeigt Mercurial leicht unterschiedliche Informationen (siehe Listing 26.25):

```

$ hg log
changeset: 1:b8dad3b41ff8
tag:      tip
user:     Dirk Deimeke <dirk@deimeke.net>

```

```
date:      Wed Jul 30 19:13:25 2014 +0200
summary:   mehr nicht
```

```
changeset: 0:f34871bf678a
user:      Dirk Deimeke <dirk@deimeke.net>
date:      Wed Jul 30 19:11:07 2014 +0200
summary:   Lies das
```

Listing 26.25 Anzeige aller Änderungen bei Mercurial

Mercurial hat ebenfalls einen sehr guten Userguide, den Sie unter der folgenden URL finden können: <https://www.mercurial-scm.org/guide>

26.2.5 Git

Git (<https://git-scm.com>) entstand ebenfalls aufgrund der Lizenzänderung von BitKeeper und wurde von Linus Torvalds ins Leben gerufen. Git ist derzeit das populärste der dezentralen (verteilten) Versionskontrollsysteme. Die folgenden Listings zeigen, dass Git als »Plappermaul« gelten kann: Keines der bisher vorgestellten Versionskontrollsysteme hat so ausführliche Ausgaben. In Git lässt sich mit einem Kommando der eigene User einstellen (siehe Listing 26.26):

```
$ git config --global user.name "Dirk Deimeke"
$ git config --global user.email "dirk@deimeke.net"
$ cat ~/.gitconfig
[user]
    name = Dirk Deimeke
    email = dirk@deimeke.net
```

Listing 26.26 Eingeben der Userinformationen bei Git

Auch in Git wird mit `init` ein neues Repository erstellt. Der Zusatz `--bare` sorgt in Listing 26.27 dafür, dass ein Repository ohne Arbeitskopie erstellt wird:

```
$ git init --bare /var/tmp/gitrepository
Initialized empty Git repository in /var/tmp/gitrepository/
$ git clone /var/tmp/gitrepository
Cloning into 'gitrepository'...
warning: You appear to have cloned an empty repository.
done.
```

Listing 26.27 Erzeugen eines Repositories mit Git

Git zeigt keine Besonderheiten, was das Hinzufügen von Dateien in Listing 26.28 angeht:

```
$ git status
# On branch master
```

```

#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#    liesmich
nothing added to commit but untracked files present (use "git add" to track)

$ git add liesmich
$ git commit -m "Lies das" liesmich
[master (root-commit) 8497685] Lies das
 1 file changed, 1 insertion(+)
 create mode 100644 liesmich
$ git push origin master
Counting objects: 3, done.
Writing objects: 100% (3/3), 218 bytes, done.
Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
To /var/tmp/gitrepository
 * [new branch]      master -> master

```

Listing 26.28 Dateien unter Versionskontrolle stellen mit Git

Der diff-Befehl wurde auch bereits vorgestellt. Seine Ausgaben unter Git sind nahezu identisch mit denen bei GNU Bazaar und Mercurial (siehe Listing 26.29):

```

$ git status
[...]
#    modified:   liesmich
#
no changes added to commit (use "git add" and/or "git commit -a")
$ git diff liesmich
diff --git a/liesmich b/liesmich
index 0abf191..21bb927 100644
--- a/liesmich
+++ b/liesmich
@@ -1,3 @@
  Lies das!
+
+Mehr habe ich nicht zu sagen.
$ git commit -m "mehr nicht" liesmich
[master f325a3a] mehr nicht
 1 file changed, 2 insertions(+)

```

```
$ git push origin master
Counting objects: 5, done.
Writing objects: 100% (3/3), 275 bytes, done.
Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
To /var/tmp/gitrepository
    8497685..f325a3a  master -> master
```

Listing 26.29 Ändern einer Datei bei Git

In Listing 26.30 zeigt sich Git sehr aufgeräumt bei der Ausgabe der Log-Informationen:

```
$ git log liesmich
commit f325a3abc2d2902e8d172140005127a4f703a2f1
Author: Dirk Deimeke <dirk@deimeke.net>
Date:   Wed Jul 30 19:18:38 2014 +0200

    mehr nicht
commit 8497685b84e0852c504578d9cf95b19b5b9fc6b9
Author: Dirk Deimeke <dirk@deimeke.net>
Date:   Wed Jul 30 19:17:40 2014 +0200
```

Lies das

Listing 26.30 Anzeige aller Änderungen bei Git

Auch für Git gibt es weiterführende Lektüre. Hier empfehlen sich das freie Pro-Git-Buch unter <https://git-scm.com/book> (englisch) bzw. das freie Git-Buch unter <https://gitbu.ch/> (deutsch).

26.3 Kommandos

In Tabelle 26.3 werden die Kommandos der einzelnen Versionskontrollsysteme einander gegenübergestellt. In den Spalten werden die Kommandos der Systeme benutzt: So steht `cvs` für Concurrent Versioning System, `svn` für Apache Subversion, `bzr` für GNU Bazaar, `hg` für Mercurial und `git` für Git.

	<code>cvs</code>	<code>svn</code>	<code>bzr</code>	<code>hg</code>	<code>git</code>
Paketname der Installation	<code>cvs</code>	<code>subversion</code>	<code>bzr</code>	<code>mercurial</code>	<code>install git</code>
Repository erstellen	<code>init</code>	<code>svnadmin create</code>	<code>init</code> <code>init-repository</code>	<code>init</code>	<code>init</code> <code>init bare</code>

Tabelle 26.3 Kommandos bei unterschiedlichen Versionskontrollsystemen

	cvs	svn	bzr	hg	git
Repository klonen	–	svnadmin hotcopy	branch	clone	clone
Arbeitskopie auschecken	checkout	checkout	checkout	clone	checkout
Hinzufügen	add	add	add	add	add
Löschen	rm	rm	rm	rm	rm
Verschieben (umbenennen)	–	mv	mv	mv	mv
Änderungen festschreiben	commit	commit	commit	commit	commit
Entfernte Änderungen übertragen	–	–	pull	pull	fetch
Lokale Änderungen übertragen	–	–	push	push	push
Arbeitskopie synchronisieren	update	update	update	pull -u	pull

Tabelle 26.3 Kommandos bei unterschiedlichen Versionskontrollsystemen (Forts.)

26.4 Serverdienste

Für das am weitesten verbreitete Versionskontrollsystem Git werden wir im Folgenden zwei Varianten betrachten, die es ermöglichen, einen eigenen Git-Server zu betreiben.

26.4.1 Git-Server mit Gitolite

Gitolite (<https://gitolite.com/>) bietet eine einfache Möglichkeit, eigene Git-Repositories zu hosten. Das Einzige, was dazu benötigt wird, ist ein Server, auf dem SSH läuft.

Gitolite installieren

Zu Beginn legen wir einen Systemuser und eine Systemgruppe namens »git« an. Die Befehle in Listing 26.31 dienen dazu als Referenz:

```
$ groupadd --system git
$ useradd --comment "Gitolite Service" --home-dir /srv/git --gid git --create-home \
  --system git
```

Listing 26.31 User und Gruppe anlegen

Sollte die Software git bis jetzt noch nicht installiert sein, dann ist jetzt ein guter Zeitpunkt, das nachzuholen. Ein Server mit Gitolite wird selbst auch mittels Git verwaltet. Dazu wird der SSH-Public-Key eines Users benötigt, der später die Repositories administrieren soll. In Listing 26.32 beispielsweise liegt der öffentliche SSH-Schlüssel des Benutzers »dirk« auf der gleichen Maschine, das muss aber nicht zwangsweise so sein.

```
$ cp /home/dirk/.ssh/id_rsa.pub /srv/git/dirk.pub
$ chown git:git /srv/git/dirk.pub
```

Listing 26.32 Öffentlichen Teil des Admin-Keys kopieren

Die Installation führen wir mit dem User »git« aus. Dazu klonen wir zuerst das Repository von GitHub (siehe Listing 26.33):

```
$ git clone https://github.com/sitaramc/gitolite.git
Cloning into 'gitolite'...
[...]
Checking connectivity... done.
```

Listing 26.33 Klonen des Repositorys

Nach dem Klonen besteht die Installation aus den drei in Listing 26.34 beschriebenen Schritten, die als User »git« ausgeführt werden müssen. Anschließend ist die Installation für den User funktionsfähig, dessen Key im vorherigen Schritt (siehe Listing 26.32) kopiert wurde.

```
$ mkdir $HOME/bin
$ gitolite/install -to $HOME/bin

$ $HOME/bin/gitolite setup -pk dirk.pub
Initialized empty Git repository in /srv/git/repositories/gitolite-admin.git/
Initialized empty Git repository in /srv/git/repositories/testing.git/
WARNING: /srv/git/.ssh missing; creating a new one
  (this is normal on a brand new install)
WARNING: /srv/git/.ssh/authorized_keys missing; creating a new one
  (this is normal on a brand new install)
```

Listing 26.34 Installation von Gitolite

Damit ist serverseitig alles getan. Die weiteren Schritte kann und muss der angelegte Admin-User per SSH und geklontem Admin-Repository durchführen.

Gitolite konfigurieren

Im Weiteren zeigen wir Ihnen, wie sich Gitolite verwalten lässt. Wir verwenden dazu einen Client, der sich auf der gleichen Maschine wie die Serverinstallation befindet. Wenn Sie das in Ihrer Umgebung ausführen wollen, verwenden Sie bitte statt *localhost* die Adresse oder den Namen Ihres Servers.



Der erste Schritt zur Verwaltung des eigenen Servers ist das Klonen des Admin-Repositorys (siehe Listing 26.35). Im Verzeichnis finden sich – bis auf die Git-Verwaltungsinformationen – nur zwei Dateien: der öffentliche Teil des Administratorschlüssels (hier im Beispiel *dirk.pub*) und eine Konfigurationsdatei mit dem Namen *gitolite.conf*.

26

```
$ git clone ssh://git@localhost/gitolite-admin.git
Cloning into 'gitolite-admin'...
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (6/6), done.
```

```
$ cd gitolite-admin
$ ls *
conf:
gitolite.conf
```

```
keydir:
dirk.pub
```

```
$ cat conf/gitolite.conf
repo gitolite-admin
  RW+   =   dirk
repo testing
  RW+   =   @all
```

Listing 26.35 Klonen des Admin-Repositorys

Wie Sie leicht feststellen können, sind in der Konfigurationsdatei zwei Repositories eingestellt: zum einen das, in dem wir uns gerade befinden, und zum anderen ein Repository namens *testing*. Weiterhin sehen Sie, dass der User *dirk* Lese-, Schreib- und Sonderrechte (»fast-forward«, »rewind« und Löschrchte auf Branches und Tags) hat und niemand anderes sonst. Die Gruppe *all* hat die gleichen Rechte auf das Repository *testing*.

Ein User in Gitolite entspricht einem öffentlichen Schlüssel, der so, wie in Listing 26.36 gezeigt, dem Repository hinzugefügt wird. Wichtig ist, dass die Namen der öffentlichen Schlüssel immer auf *.pub* enden.

```
$ cp /tmp/paul.pub keydir
$ git add keydir/paul.pub
$ git commit -m "User Paul added"
[master a741b6b] User Paul added
 1 file changed, 1 insertion(+)
 create mode 100644 keydir/paul.pub
$ git push
Counting objects: 6, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 676 bytes | 0 bytes/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To ssh://git@localhost/gitolite-admin.git
   5e8f3ee..a741b6b  master -> master
```

Listing 26.36 Hinzufügen eines neuen Users

Das Anlegen einer Gruppe erfolgt ähnlich einfach. Gruppen beginnen bei Gitolite mit einem »@«-Zeichen, und in Listing 26.37 legen wir die Gruppe `adminbuch` mit den Mitgliedern `dirk` und `paul` an und zusätzlich je eine Gruppe, die nur einen User enthält.

```
$ head -3 conf/gitolite.conf
@dirk      = dirk
@paul      = paul
@adminbuch = paul @dirk
```

Listing 26.37 Anlegen einer Gruppe

Wie Sie sehen, können Gruppen sowohl andere Gruppen wie auch »Personen« enthalten. Mitglieder werden durch Leerzeichen getrennt. Wir empfehlen Ihnen, aus Gründen der Übersichtlichkeit möglichst immer Gruppen einzusetzen. Nach einem `git commit` und `git push` ist diese neue Konfiguration aktiv. Ein neues Repository können Sie nach dem Muster der bestehenden Repositories anlegen. Listing 26.38 zeigt Ihnen, wie das geht:

```
$ tail -2 conf/gitolite.conf
repo adminbuch
  RW+      = @adminbuch
$ git commit -am "New repository"
[master dc58e17] New repository
 1 file changed, 6 insertions(+), 3 deletions(-)
$ git push
Counting objects: 7, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 406 bytes | 0 bytes/s, done.
```

```
Total 4 (delta 0), reused 0 (delta 0)
remote: Initialized empty Git repository in /srv/git/repositories/adminbuch.git/
To ssh://git@localhost/gitolite-admin.git
    f04e394..dc58e17  master -> master
```

Listing 26.38 Hinzufügen eines neuen Repositorys

Sobald Sie die veränderte Datei zum Server übertragen haben, gibt es ein neues Repository namens `adminbuch`, für das die User `dirk` und `paul` als Mitglieder der Gruppe `adminbuch` volle Rechte haben.

Sollten Sie dem User `paul` erlauben wollen, lesend auf das Admin-Repository zuzugreifen, fügen Sie analog zu Listing 26.39 eine Zeile für die Leseberechtigungen ein:

```
$ grep -A 2 gitolite-admin conf/gitolite.conf
repo gitolite-admin
    RW+    =   dirk
    R      =   paul
```

Listing 26.39 Vergabe von Lese- und Schreibberechtigungen

Selbstverständlich können Sie anstelle der User auch die entsprechenden Gruppen verwenden. Gitolite bietet Ihnen noch viele weitere Möglichkeiten, Repositories granular zu konfigurieren und beispielsweise serverseitige Hooks einzusetzen (und vieles andere mehr). Das zu beschreiben, würde aber den Rahmen dieses Kapitels bei Weitem sprengen, bitte konsultieren Sie dazu die Homepage des Projekts: <https://gitolite.com>

26.4.2 Git-Server mit Gitea

Gitea (<https://gitea.io/>) bietet eine einfache Möglichkeit, eigene Git-Repositories zu hosten. Als Besonderheit bietet Gitea eine Weboberfläche, die der von GitHub (<https://github.com/>) nachempfunden ist. Repositories können nicht nur via SSH, sondern auch via HTTP oder HTTPS benutzt werden.

Die Quelltexte des vor Ihnen liegenden Buches werden ebenfalls auf einem Server mittels Gitea verwaltet.



Gitea installieren

Analog zu unserem Vorgehen bei Gitolite legen wir zu Beginn einen Systemuser und eine Systemgruppe namens `gitea` an:

```
$ groupadd --system gitea
$ useradd --comment "Gitea Service" --home-dir /srv/gitea --gid gitea --create-home \
    --system gitea
```

Listing 26.40 User und Gruppe anlegen



Bitte beachten Sie, dass Sie den User natürlich beliebig benennen können. Sollten Sie Gitea nicht verwenden, spricht gar nichts dagegen, den User »git« zu nennen.

Selbstverständlich benötigen Sie für einen Server, der Git-Repositories verwaltet, auch die Software `git`. Bitte installieren Sie Git mit den Mitteln Ihres Betriebssystems. Zur Installation von Gitea laden Sie sich die entsprechende Binärdatei von der Download-Seite des Projekts <https://dl.gitea.io/gitea/> herunter. In Listing 26.41 finden Sie die Befehle, die Sie benötigen, um die zum Druckzeitpunkt aktuelle Version von Gitea auf Ihrem Server zu installieren:

```
$ sudo -u gitea -i
$ mkdir gitea
$ cd gitea
$ curl -LO https://dl.gitea.io/gitea/1.20/gitea-1.20-linux-amd64
[...]
$ chmod 750 gitea-1.20-linux-amd64
$ ln -s gitea-1.20-linux-amd64 gitea
```

Listing 26.41 Installation von Gitea

Der letzte Befehl, das Setzen des Links, wäre nicht unbedingt nötig, erleichtert aber bei künftigen Versionen die Aktualisierung. Starten Sie Gitea einmalig, um fehlende Dateien und Verzeichnisse zu erstellen:

```
$ ./gitea web
2023/05/22 13:11:09 [W] Custom config '/srv/gitea/gitea/custom/conf/app.ini' \
      not found, ignore this if you're running first time
2023/05/22 13:11:09 [T] AppPath: /srv/gitea/gitea/gitea
2023/05/22 13:11:09 [T] AppWorkPath: /srv/gitea/gitea
2023/05/22 13:11:09 [T] Custom path: /srv/gitea/gitea/custom
2023/05/22 13:11:09 [T] Log path: /srv/gitea/gitea/log
2023/05/22 13:11:09 [I] Log Mode: Console(Info)
2023/05/22 13:11:09 [I] XORM Log Mode: Console(Info)
2023/05/22 13:11:09 [I] Cache Service Enabled
2023/05/22 13:11:09 [I] Session Service Enabled
2023/05/22 13:11:09 [I] SQLite3 Supported
2023/05/22 13:11:09 [I] Run Mode: Development
2023/05/22 13:11:09 [I] Gitea v1.5.1 built with: bindata, sqlite
2023/05/22 13:11:09 [I] Listen: http://0.0.0.0:3000
2023/05/22 13:11:09 Serving [::]:3000 with pid 17937
```

Listing 26.42 Erster Start von Gitea

Rufen Sie nun die URL <http://localhost:3000/install> einmal auf, um die Grundkonfiguration zu erstellen. Als Datenbank für Tests können Sie SQLite verwenden. Falls Sie ein größeres Setup erstellen wollen, empfehlen wir eine Verbindung zu einem »echten« Datenbanksystem.

tem«. Vergessen Sie dabei nicht, einen Admin-Usernamen und ein Passwort zu vergeben. Beenden können Sie den Server mit der Tastenkombination `Strg` + `C`.

Gitea konfigurieren

Ein rudimentäres Beispiel für eine *systemd-service*-Datei finden Sie in Listing 26.43. Erstellen Sie eine Datei */etc/systemd/system/gitea.service* mit dem folgenden Inhalt:

```
[Unit]
Description=Gitea (Git with a cup of tea)
After=syslog.target
After=network.target

[Service]
Type=simple
User=gitea
Group=gitea
WorkingDirectory=/srv/gitea/gitea
ExecStart=/srv/gitea/gitea/gitea web
Restart=always
Environment=USER=gitea HOME=/srv/gitea

[Install]
WantedBy=multi-user.target
```

Listing 26.43 »systemd-service«-Datei für Gitea

Um diesen Dienst zu aktivieren, sollten Sie einmal als *root* den Befehl `systemd daemon-reload` ausführen. Anschließend können Sie mit `systemctl start gitea` den Dienst starten. Weitere Konfigurationsmöglichkeiten mit Bezug zu *systemd* sind auf der offiziellen Webseite unter <https://docs.gitea.io/en-us/linux-service/> beschrieben.

Da der Zugriff auf die Gitea-Instanz unverschlüsselt via HTTP auf Port 3000 erfolgt, wäre es sinnvoll, einen Webserver oder Squid als Reverse-Proxy vor die Gitea-Instanz zu stellen. Der Webserver würde in diesem Fall die eingehenden Anfragen auf Port 443 (HTTPS) an Gitea weiterleiten. Das können Sie dadurch erreichen, dass Sie beispielsweise die folgenden Zeilen aus Listing 26.44 der Konfigurationsdatei Ihres *VirtualHosts* hinzufügen:

```
ProxyPreserveHost On
ProxyPass / http://localhost:3000/
ProxyPassReverse / http://localhost:3000/
```

Listing 26.44 Reverse-Proxy mit Apache

Passen Sie in der Datei */srv/gitea/gitea/custom/conf/app.ini* die Einträge im Abschnitt `[server]` entsprechend an. Im Beispiel verwenden wir *gitea.example.com* (siehe Listing 26.45):

```
[server]
SSH_DOMAIN      = gitea.example.com
HTTP_PORT       = 3000
ROOT_URL        = https://gitea.example.com/
DISABLE_SSH     = false
SSH_PORT        = 22
LFS_START_SERVER = false
OFFLINE_MODE    = false
```

Listing 26.45 Gitea-Server-Konfiguration

Alle Möglichkeiten, die Gitea Ihnen bietet, können wir leider in diesem Kapitel nicht beschreiben. Daher möchten wir Ihnen die sehr umfangreiche Dokumentation unter <https://docs.gitea.io/en-us/> ans Herz legen.

TEIL VI
Automatisierung

Kapitel 27

Scripting

In diesem Kapitel bekommen Sie einige wertvolle Tipps und Hinweise zur Kommandozeile und zum Scripting.

27

Was dem Maurer die Kelle oder dem Zimmermann der Hammer, das ist dem Sysadmin die Shell: das ultimative Werkzeug des Alltags. Der richtige Umgang mit den von der Shell zur Verfügung gestellten Mitteln erleichtert dem Sysadmin nicht nur die Arbeit, sondern kann ihm diese auch durch automatisierte Skripte vollständig abnehmen.

In diesem Kapitel erfahren Sie einiges zum Umgang mit Skripten, wie Sie sie effizienter gestalten und wie Sie sie einsetzen, um den Arbeitsalltag einfacher zu bewältigen.

27.1 Aufgebohrte Muscheln

In den Werkzeugkasten jedes Administrators gehören das Basiswissen über Scripting und – da wir uns auf Linux-Systemen befinden – vor allem Kenntnisse in der Shell- und Python-Programmierung. Perl hat an Bedeutung verloren, und obwohl Microsoft ihre PowerShell vor einigen Jahren auch auf Linux- und macOS-Systemen portiert hat, ist die Verbreitung abseits der Windows-Welt gering. Viele Skripte werden mittlerweile in Python geschrieben, aber wenn Sie systemnah mit Linux-Maschinen arbeiten müssen, ist die `bash` noch immer das beste Werkzeug. Grundsätzlich ist es keine schlechte Idee, sich seine Arbeits- und Skriptumgebung so zusammenzustellen, wie es am besten passt. Dabei sollten Sie allerdings auf ein paar Stolpersteine achten:

- ▶ Kann ich das, was ich in meiner Arbeitsumgebung erfolgreich getestet habe, auch auf den anderen Systemen und mit anderen (technischen) Benutzern im Haus laufen lassen?
- ▶ Welche Vorbedingungen (Programme, Versionsstände, Umgebungsvariablen) benötigen meine Skripte, um auf anderen Umgebungen lauffähig zu sein?

Weiterhin ist es sinnvoll, sprechende Namen für Variablen und Funktionen zu verwenden und nicht zu versuchen, den Preis für den Code zu gewinnen, der die Funktion des Skripts am besten verschleiert (siehe »Obfuscated-Perl-Contest« in Wikipedia, https://de.wikipedia.org/wiki/Obfuscated_Perl_Contest). Aus diesem Grund werden die folgenden Beispiele und Anregungen alle mit der `bash` und den Tools getestet, die serienmäßig in-

stalliert sind. Damit stellen wir sicher, dass die Skripte und Kommandozeilen auf allen Systemen ohne weitere Änderung funktionieren.



Viele Skripte verweisen im *Shebang* (`#!`) am Anfang eines Skripts auf `/bin/sh`. Dass das nicht immer richtig ist, zeigen einige Skripte von Fremdanbietern. `/bin/sh` ist auf fast allen Linux-Systemen ein Link auf eine andere Shell. Bei Debian und Ubuntu wird dort die *Almquist Shell* (*dash*) gelinkt, bei CentOS und openSUSE ist es die *GNU Bourne Again Shell* (*bash*). Aus diesem Grund sollte immer die Shell angegeben werden, mit der das Skript auch getestet wurde, und nicht die, die der Link bereitstellt.

27.2 Vom Suchen und Finden: ein kurzer Überblick

Zu den Hauptaufgaben der Systemadministration gehören das Durchforsten von Log-Dateien genauso wie das regelmäßige Extrahieren von Informationen und das Erstellen von Reports aus diesen Informationen. Daher nimmt das Suchen und Verarbeiten von Textdateien einen wichtigen Stellenwert innerhalb der täglichen Arbeit ein.

27.2.1 Die Detektive: *grep*, *sed* und *awk*

Zu den wichtigsten Tools gehören *grep*, *sed* und *awk*, die manchmal nach den Anfangsbuchstaben unter dem Kürzel *GAS* zusammengefasst werden.

Die Hauptaufgabe von *grep* besteht im Durchsuchen von Textdateien und in der Ausgabe der Zeilen, die einen Treffer enthalten. Für die Herkunft des Begriffs *grep* gibt es zwei Erklärungsansätze: Zum einen soll es für *global/regular expression/print* stehen und zum anderen für *global search for a regular expression and print out matched lines*. Bei *sed* (den *Stream Editor*) liegt der Fokus auf der zeilenweisen Veränderung von Textzeilen.

awk, dessen Name sich aus den Anfangsbuchstaben der Nachnamen seiner drei Autoren zusammensetzt – Alfred V. Aho, Peter J. Weinberger und Brian W. Kernighan –, ist das mächtigste der drei Tools. *awk* bringt eine eigene Programmiersprache mit. Allerdings wird es häufig nur dazu verwendet, Teile einer Zeile formatiert auszugeben, eine Aufgabe, die auch *cut* erfüllen könnte. *awk* beherrscht alles, was *sed* und *grep* können, ist allerdings etwas schwieriger zu lernen. Obwohl Perl an Bedeutung verloren hat, ist es ungeschlagen bei Suchvorgängen, die sich im »Kommandozeilenmodus« einfach und schnell erledigen lassen. Nehmen wir das folgende Beispiel:

```
dirk@example:/# ps -ef | grep ssh | grep -v grep
```

Listing 27.1 Laufende »ssh«-Prozesse finden: »grep«

Dieser Befehl listet alle Prozesse auf, bei denen das Wort *ssh* in der Ausgabe von *ps -ef* zu finden ist. Das *grep -v grep* am Ende sorgt dafür, dass der Suchprozess sich nicht selbst

findet. Wir haben also einen `ps`-Aufruf und zwei `grep`-Aufrufe. Mit `awk` reduziert sich der Aufwand auf einen `ps`-Aufruf und einen `awk`-Aufruf, wie dieser Befehl zeigt:

```
dirk@example:/# ps -ef | awk '/ssh/ && ! /awk/ {print}'
```

Listing 27.2 Laufende »ssh«-Prozesse finden: »awk«

Die Befehle in den Listings 27.1 und 27.2 sind nicht nur ein Beispiel dafür, dass viele Wege nach Rom führen, sondern vor allem auch dafür, dass Sie Ihre Tools gut kennen sollten. Meist ist für die Lösung einer Problemstellung nur ein Tool nötig, und Zusatztools sind überflüssig (ein Paradebeispiel dafür ist `cat datei.txt | grep XYZ`).

Wie jeder gute Handwerker müssen Sie dafür Ihr Werkzeug gut beherrschen. Ein Blick in die Manpages eines Tools eröffnet einem Sysadmin meist ungeahnte Möglichkeiten. Scheuen Sie sich also nicht, auch einmal nachzulesen, wie mächtig die von Ihnen eingesetzten Programme sind.

27

27.2.2 Reguläre Ausdrücke verstehen und anwenden

Für viele sind reguläre Ausdrücke ein Buch mit sieben Siegeln. Dabei sind sie eines der besten Werkzeuge, um effiziente Skripte zu erstellen. Bei regulären Ausdrücken, die oft auch als *Regex*¹ bezeichnet werden, handelt es sich um syntaktische Regeln zur Beschreibung von Mengen oder Untermengen einer Zeichenkette. Mit diesen Regeln können Zeichenketten extrahiert werden (Mustersuche oder *Pattern Matching*²), auch wenn deren genaue Abfolge nicht bekannt ist. Ein weiterer Anwendungszweck ist das Suchen und Ersetzen, das mit regulären Ausdrücken ebenfalls vereinfacht werden kann.

Viele Programme verwenden eigene Implementierungen von regulären Ausdrücken, die sich in Umfang und Syntax unterscheiden. Die folgenden drei großen decken fast alle zu findenden Implementierungen ab:

- ▶ **Basic Regular Expressions (BRE)**
Dies sind nach dem *POSIX*-Standard definierte reguläre Ausdrücke – sie finden zum Beispiel beim Editor *vim* Anwendung.
- ▶ **Extended Regular Expressions (ERE)**
Ebenfalls nach dem *POSIX*-Standard definierte reguläre Ausdrücke mit erweitertem Funktionsumfang – sie finden zum Beispiel bei *egrep* Anwendung.
- ▶ **Perl Compatible Regular Expressions (PCRE)**
Reguläre Ausdrücke nach dem *PCRE*-Standard orientieren sich an der Implementierung von *Regex* in der Programmiersprache *Perl*.

1 *regular expression*, engl. für *regulärer Ausdruck*.

2 *to match*, mit etwas übereinstimmen.

Die Implementierungen unterscheiden sich vor allem in der Syntax. Bei *BREs* muss die Sonderfunktion von Spezialzeichen durch einen umgekehrten Schrägstrich – *Backslash* (\) – auch innerhalb der Syntax aufgehoben werden, zum Beispiel: `s\{2\}`. Dies ist hingegen bei *EREs* nicht notwendig, zum Beispiel: `s{2}`.

Bei regulären Ausdrücken nach dem *PCRE*-Standard wird der erste Rückbezug durch `$1` angegeben, ansonsten werden diese mit einem führenden Backslash (\) dargestellt.

BRE	ERE	PCRE	Bedeutung
ab	ab	ab	die Zeichenkette »ab« in genau dieser Reihenfolge
.	.	.	ein beliebiges Zeichen (bei Dateinamenexpansion das »?«)
z*	z*	z*	beliebig oft »z«, auch kein Vorkommen
x\+	x+	x+	mindestens einmal »x«
a\{6\}	a{6}	a{6}	genau sechsmal »a«
a\{3,\}	a{3,}	a{3,}	mindestens dreimal »a«
a\{2,4\}	a{2,4}	a{2,4}	zwei- bis viermal »a«
[abc]	[abc]	[abc]	ein einzelnes »a«, »b« oder »c«
[a-z]	[a-z]	[a-z]	ein beliebiges Zeichen zwischen »a« und »z«
[^ab]	[^ab]	[^ab]	ein beliebiges Zeichen, ohne »a« und ohne »b«
y\?	y?	y?	einmal oder keinmal »y«
\(...\)	(...)	(...)	Mit Klammern werden Ausdrücke für Rückbezüge gruppiert.
a\b	a b	a b	entweder »a« oder »b«
^	^	^	Zeilenanfang
\$	\$	\$	Zeilenende
ab	ab	ab	die Zeichenkette »ab« in genau dieser Reihenfolge
\<	\<	\<	Anfang eines Worts
\>	\>	\>	Wortende
\w	\w	\w	alphanumerische Zeichen und der Unterstrich (Wortzeichen)

Tabelle 27.1 Syntax der regulären Ausdrücke

BRE	ERE	PCRE	Bedeutung
\w	\w	\w	kein Wortzeichen
\d	\d	\d	eine Ziffer
\D	\D	\D	keine Ziffer
\s	\s	\s	Leerzeichen (<i>Whitespace</i>)
\S	\S	\S	kein Leerzeichen

Tabelle 27.1 Syntax der regulären Ausdrücke (Forts.)

Tabelle 27.1 zeigt die Unterschiede in der Syntax bei den einzelnen Implementierungen von regulären Ausdrücken. Wenn mit regulären Ausdrücken nach Sonderzeichen gesucht werden soll, muss deren Sonderfunktion mit einem Backslash (\) aufgehoben werden. Dies darf aber nicht mit der Dateinamensexpansion (*Globbing*) verwechselt werden, da hier eine andere Syntax angewandt wird. Ein zentrales Element von regulären Ausdrücken sind Rückbezüge. Mit diesen können nicht nur Zeichenketten extrahiert, ein Suchen und Ersetzen durchgeführt, sondern auch Vergleiche vorgenommen werden. Für das Durchsuchen eines Wörterbuchs sei folgende Aufgabenstellung vorausgesetzt:

Suchen Sie nach allen Begriffen, die aus fünf Buchstaben bestehen und von vorn und hinten gelesen gleich aussehen, wie beispielsweise »rotor«.

Dies kann sowohl über `egrep` als auch mit *Perl* erreicht werden:

```
### egrep
d@example:/# egrep "^(..)(..)\2\1$" /usr/share/dict/american-english
```

```
### Perl
d@example:/# perl -ne 'print if /^(..)(..)\2\1$/' /usr/share/dict/american-english
```

Listing 27.3 Suche und Rückgabe mit »`egrep`«

Beide Befehle liefern dasselbe Ergebnis zurück. In den aktuellen Versionen von *grep* können auch unterschiedliche reguläre Ausdrucksarten angewandt werden. Mit `egrep` (oder auch `grep -E`) wird ein ERE verwendet. Mit dem Aufruf von `grep -P` wird ein PCRE benutzt. Je nach Art des regulären Ausdrucks ist *grep* mit EREs um den Faktor 2 bis 10 langsamer als mit PCREs. Auch Linux-Shells, etwa die *bash* ab Version 3, verstehen reguläre Ausdrücke, sodass Sie diese direkt in Skripten verwenden können. Listing 27.4 zeigt ein Beispiel für einen *bash*-Regex:

```
#!/bin/bash
FILENAME=dh-20091005-ausgabe-006.ogg
REGEX='^dh-([0-9]{4})([0-9]{2})([0-9]{2})-ausgabe-([0-9]{3}).ogg$'
```

```


if [[ $FILENAME =~ $REGEX ]]
then
    jahr=${BASH_REMATCH[1]}
    monat=${BASH_REMATCH[2]}
    tag=${BASH_REMATCH[3]}
    episode=${BASH_REMATCH[4]}
fi
echo "Jahr: $jahr, Monat: $monat, Tag: $tag, Episode: $episode"

```

Listing 27.4 Beispiel-»Regex« in einem »bash«-Skript

Hier wird über den regulären Ausdruck REGEX angegeben, aus welchen Teilen FILENAME besteht. Die *if*-Abfrage wendet den regulären Ausdruck aufgrund des Vergleichsoperators =~ an. Die Variablen werden über das *bash*-interne Array BASH_REMATCH den einzelnen Rückbezügen zugewiesen.

27.3 Fortgeschrittene Shell-Programmierung

Ein wahres Luxusgut der Linux-Shells ist die Expansion. Diese haben die meisten Sysadmins im alltäglichen Leben bereits durch die Dateinamen- oder Kommandoexpansion mittels Eingabe von  lieb gewonnen.

27.3.1 Expansionsschemata

Die Expansion spielt auch für die Erstellung von Skripten eine große Rolle. Sie folgt nämlich einem festgelegten Schema. Brechen Sie dieses, ist Ihr Skript nicht lauffähig. Die Reihenfolge der Expansion entspricht folgendem Ablauf:

»brace expansion«

Dies ist die Expansion der Inhalte, die von Klammern umschlossen sind.

Expansion	Bedeutung
{a,b}	wird als »a b« expandiert.
x{a,b}y	wird als »xay xby« expandiert.
{1..5}	wird als »1 2 3 4 5« expandiert.
{1..5..2}	wird als »1 3 5« expandiert. Syntax: {<START>..<END>..<INKREMENT>}

Tabelle 27.2 Klammersnexpansion – »brace expansion«

Expansion	Bedeutung
{a..d}	wird als »a b c d« expandiert.
M{a,e}{i,y}er	wird als »Maier Mayer Meier Meyer« expandiert.

Tabelle 27.2 Klammerexpansion – »brace expansion« (Forts.)

Die Expansion funktioniert ähnlich wie bei Pfaden und Dateien, nur dass diese nicht existieren müssen. Die Auswertung erfolgt von links nach rechts, dabei dürfen sowohl vor als auch hinter den Klammern Inhalte stehen. Tabelle 27.2 zeigt eine Übersicht über die gängige Verwendung von Klammerexpansionen.

In Listing 27.5 finden Sie ein Paradebeispiel zur Benutzung von Klammerexpansionen – die Erstellung von inkrementierten Verzeichnissen (CD01 bis CD09):

```
dirk@example:~/Musik/4D5/# mkdir CD0{1..9}
```

Listing 27.5 Beispiel für die Klammerexpansion: »mkdir«

tilde expansion

Die Tilde hat auf Linux-Systemen eine Sonderfunktion. Mit ihr können die in Tabelle 27.3 aufgeführten Verzeichnisse direkt angesprochen werden.

Art	Bedeutung
~loginname	enthält das Homeverzeichnis des angegebenen Benutzers – ist loginname nicht gesetzt, wird das Homeverzeichnis des aktuellen Benutzers expandiert.
~+	enthält das aktuelle Arbeitsverzeichnis.
~-	enthält das vorherige Arbeitsverzeichnis.

Tabelle 27.3 Tildenexpansion

parameter and variable expansion

Diese Art der Expansion dient zur Entwicklung von Parametern und Variablen. Das Dollarzeichen (\$) leitet eine Parameterexpansion, eine Kommandoersetzung oder die Auswertung von arithmetischen Ausdrücken ein. Dazu zählt auch die String-Verarbeitung. Tabelle 27.4 zeigt einen Auszug der möglichen Expansionen.

Tabelle 27.4 zeigt zudem zwei Beispiele für die Parameterentwicklung. Weitere Beispiele finden Sie in Abschnitt 27.3.3 unter dem Stichpunkt »String-Verarbeitung«.

Art	Bedeutung
<code>\${parameter}</code>	enthält den Inhalt der Variablen <code>parameter</code> .
<code>\${parameter:-word}</code>	enthält den Inhalt der Variablen <code>parameter</code> . Wenn <code>parameter</code> nicht gesetzt oder null ist, wird <code>word</code> eingesetzt.

Tabelle 27.4 Parameter- und Variablenexpansion

arithmetic expansion

Ausführung von Rechenoperationen innerhalb der Shell, zum Beispiel `$((2 + 2))`. Dabei können die Operationen durchgeführt werden, die in Tabelle 27.5 aufgelistet sind.

Operator	Bedeutung
<code>id++</code> , <code>id--</code>	Postinkrement und Postdecrement der Variablen <code>id</code>
<code>++id</code> , <code>--id</code>	Preinkrement und Predecrement der Variablen <code>id</code>
<code>+</code> , <code>-</code>	ganzzahlige Addition und Subtraktion
<code>*</code> , <code>/</code> , <code>%</code>	Multiplikation, Division und Rest
<code>**</code>	Potenzieren
<code>-</code> , <code>+</code>	unäres Minus und Plus
<code>!</code> , <code>~</code>	logische und bitweise Negation
<code><<</code> , <code>>></code>	bitweise Verschiebung nach links und rechts
<code>==</code> , <code>!=</code>	gleich und ungleich
<code><=</code> , <code>>=</code> , <code><</code> , <code>></code>	Vergleichsoperatoren
<code>&</code>	bitweise UND-Verknüpfung
<code>^</code>	bitweise exklusive ODER-Verknüpfung (XOR)
<code> </code>	bitweise ODER-Verknüpfung
<code>&&</code>	logische UND-Verknüpfung
<code> </code>	logische ODER-Verknüpfung

Tabelle 27.5 Operationen

Operator	Bedeutung
<code>expr ? expr : expr</code>	Zustandsoperator
<code>=, *=, /=, %=, +=, -=, <<=, >>=, &=, ^=, =</code>	Zuweisungen

Tabelle 27.5 Operationen (Forts.)

command substitution

Die Kommandosubstitution erfolgt ebenfalls von links nach rechts. Es werden zwei Arten der Kommandosubstitution unterschieden:

► **new-style**

Hier wird die Substitution mit einem vorangestellten Dollarzeichen (\$) in Klammern gesetzt: `$(...)`.

► **old-style: »backticks«**

Hier wird die Substitution durch sogenannte *backticks* (``) umschlossen.

Wir empfehlen Ihnen, den Klammerausdruck für die Kommandosubstitution zu verwenden, da Sie damit auch mehrere Aufrufe ineinander verschachteln können. Für *backticks* gelten Sonderregeln, die vor allem beim Verketteten von Ausdrücken zum Tragen kommen.

Geschwindigkeitssteigerung

Der Ausdruck `$(cat datei.txt)` ist deutlich langsamer als das Ausgeben des Inhalts der Datei `datei.txt` mit den *bash*-Bordmitteln. Ersetzen Sie daher solche Ausdrücke durch folgende Syntax: `$(< datei.txt)`



word splitting

Eine Worttrennung wird nur auf expandierten Werten durchgeführt, die nicht von einfachen oder doppelten Anführungszeichen umschlossen sind. Dabei werden Wörter in Zeilen getrennt, wenn diese durch ein *space*, *tab* oder *newline* getrennt sind. Anhand welcher Zeichen diese Trennung vorgenommen wird, definiert der IFS (*Internal Field Separator*, engl. für *internes Feldtrennzeichen*).

pathname expansion

Abschließend werden Datei- und Verzeichnisnamen expandiert (*Globbering*). In eckigen Klammern können Zeichenklassen angegeben werden. Abbildung 27.1 zeigt, welche Auswahl dabei möglich ist.

Posix Character Classes								
print				space		cntrl ❶	xdigit	
<space>	graph			punct ❸	blank	\r \n \v \t	00-1F 7F	[0-9] [a-f] [A-F]
	alnum ❷							
	alpha		digit ❷					
	upper	lower	[0-9]					
	[A-Z]	[a-z]						
			21-2F 3A-40 5B-60 7B-7E	<space> \t				

❶ Alle Steuerzeichen, inklusive \o \a \b \t \n \v \f \r und

❷ Fixiert und nicht erweiterbar: alnum = alpha + digit; digit = [0-9]

❸ Sonderzeichen: ! ~ # \$ % ^ () * + , - . / : ; < = > ? @ [\] ^ _ ` { | } ~

Abbildung 27.1 POSIX-Zeichenklassen

Die Sonderzeichen aus Tabelle 27.6 können ebenfalls eingesetzt werden.

Sonderzeichen	Bedeutung
*	jede beliebige Zeichenfolge
?	ein beliebiges Zeichen
[...]	jedes der eingeschlossenen Zeichen

Tabelle 27.6 Datei- und Verzeichnisnamenexpansion

Als letzter Expansionschritt werden noch alle Anführungszeichen und Backslashes entfernt, die nicht das Ergebnis einer Expansion waren.

27.3.2 Umgebungsvariablen

Wie in jeder Programmiersprache stehen Ihnen in einer Shell auch Variablen zur Verfügung. Diese haben einen definierten Gültigkeitsbereich, in dem sie verwendet werden können. Programme, die Sie innerhalb einer Shell oder eines Shell-Skripts starten, werden in einer Sub-Shell gestartet. Diese bekommt das *Environment* der ursprünglichen Shell vererbt. Das Environment können Sie mit dem Befehl `env` anzeigen lassen. Die im Environment definierten Variablen können Sie direkt über die Shell verarbeiten. So erhalten Sie beim Ausführen des Befehls `echo $USERNAME` den Inhaber der Shell als Ausgabe. Damit Programme auf selbst

definierte Variablen zugreifen können, müssen diese im Environment definiert sein. Ansonsten ist den Programmen in der Sub-Shell die Variable nicht bekannt. Abbildung 27.2 zeigt das Verhalten vor und nach dem Programmaufruf.

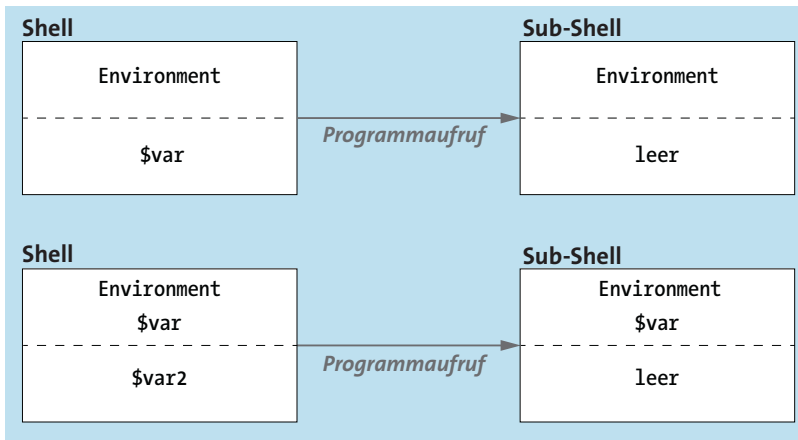


Abbildung 27.2 Variablenvererbung

Im ersten Beispiel von Abbildung 27.2 besitzt die Variable `$var` nur eine lokale Gültigkeit und ist somit nur in der ursprünglichen Shell verfügbar. Im zweiten Beispiel hingegen wurde die Variable `$var` dem Environment hinzugefügt und ist somit auch in der Sub-Shell verfügbar. Zusätzlich gibt es noch einen Spezialfall. Wenn Sie die Variablenzuweisung und das Kommando innerhalb einer Zeile ausführen, ist die Variable für das Kommando verfügbar. Um die Variable `$var` dem Environment hinzuzufügen, können Sie diese Methoden anwenden:

- ▶ **export**
Über den Befehl `export <VAR>` werden Variablen dem Environment hinzugefügt.
- ▶ **source**
Der Befehl `source <FILENAME>` exportiert Variablenzuweisungen aus einer Datei. Dies können Sie auch über die Syntax `. <FILENAME>` erreichen.

27.3.3 »Back to bash«: ein tieferer Blick in die Muschel

Die Standard-Shell `bash` verfügt über eine Vielzahl von Möglichkeiten, die vielen Benutzern gar nicht bekannt sind. Auch dem routinierten Sysadmin wird die `bash` immer wieder kleine Geheimnisse offenbaren. Einige davon möchten wir Ihnen nun etwas näher bringen.

Spezialparameter

Die `bash` hält viele Informationen über Spezialparameter bereit. Tabelle 27.7 enthält die Spezialparameter der `bash`.

Parameter	Bedeutung
\$*	alle übergebenen Parameter in einem Wort, durch Feldseparator getrennt
\$@	alle übergebenen Parameter jeweils in einem Wort
\$#	Anzahl der Parameter
\$?	Rückgabewert (Status) des zuletzt ausgeführten Kommandos
\$-	steht für die Optionsflags (von <i>set</i> oder aus der Kommandozeile)
\$\$	Prozessnummer der Shell
#!	Prozessnummer des zuletzt im Hintergrund aufgerufenen Kommandos
\$0	Name des Shell-Skripts
\$10	zehnter übergebener Parameter
\$_	letztes Argument des zuletzt ausgeführten Kommandos

Tabelle 27.7 »bash«-Spezialparameter

String-Verarbeitung

Wie bereits erörtert wurde, ist das Durchsuchen und Extrahieren von Informationen aus Log-Daten eine häufige Aufgabe des Sysadmins. Dazu ist es oft notwendig, die Daten weiterzuverarbeiten, damit sie entweder besser dargestellt werden können oder um sie in ein anderes Format zu konvertieren, das besser verarbeitet werden kann. Für diesen Anwendungszweck stellt uns die Shell eine Vielzahl von Verarbeitungsmöglichkeiten für Strings zur Verfügung. Tabelle 27.8 zeigt die Möglichkeiten der String-Verarbeitung in der Shell.

Variable	Bedeutung
\${str:pos}	extrahiert von <i>str</i> ab Position <i>pos</i> .
\${str:pos:len}	extrahiert mit der Länge von <i>len</i> ab Position <i>pos</i> von <i>str</i> .
\${str#sub}	löscht den kürzesten Treffer von <i>sub</i> am Beginn von <i>str</i> .
\${#}str	enthält die Länge von <i>str</i> .
\${str##sub}	löscht den längsten Treffer von <i>sub</i> am Beginn von <i>str</i> .
\${str%sub}	löscht den kürzesten Treffer von <i>sub</i> am Ende von <i>str</i> .

Tabelle 27.8 String-Verarbeitung in der »bash«

Variable	Bedeutung
<code>\${str%sub}</code>	löscht den längsten Treffer von <code>sub</code> am Ende von <code>str</code> .
<code>\${str/sub/rep}</code>	ersetzt den ersten Treffer von <code>sub</code> durch <code>rep</code> in <code>str</code> .
<code>\${str//sub/rep}</code>	ersetzt alle Treffer von <code>sub</code> durch <code>rep</code> in <code>str</code> .
<code>\${str/#sub/rep}</code>	wenn <code>sub</code> am Anfang von <code>str</code> ist, <code>sub</code> durch <code>rep</code> ersetzen.
<code>\${str/%sub/rep}</code>	wenn <code>sub</code> am Ende von <code>str</code> ist, <code>sub</code> durch <code>rep</code> ersetzen.

Tabelle 27.8 String-Verarbeitung in der »bash« (Forts.)

Lokale Variablen

Die richtige Behandlung von Variablen ist ein weiterer wichtiger Punkt bei der Erstellung von Shell-Skripten. Dank der automatischen Typzuweisung der *bash* ist eine Deklaration von Variablen nicht zwingend notwendig. Die *bash* fasst jede Variable erst einmal als String auf. Je nach Kontext kann die Variable auch als Integer interpretiert werden. Zusätzlich haben Variablen, wie bereits in Abschnitt 27.3.2, »Umgebungsvariablen«, erörtert wurde, einen Gültigkeitsbereich. Damit Variablen nur innerhalb einer Funktion gültig sind, müssen sie explizit als lokale Variablen definiert werden. Verwenden Sie dafür das `local`-Kommando:

```
local -i VAR
```

Listing 27.6 Lokale Definition der Variablen »VAR« als Integer mit »local«

Das Kommando `local` kann dabei die Parameter aus Tabelle 27.9 verarbeiten, um Variablen einen Typ fest zuzuweisen bzw. diese gesondert zu behandeln.

Parameter	Bedeutung
<code>-a</code>	Variablen sind Arrays.
<code>-f</code>	Funktionsnamen
<code>-i</code>	Variablen werden als Integer behandelt.
<code>-r</code>	Variablen sind <i>readonly</i> = Konstanten.
<code>-t</code>	Namen bekommen das <i>Trace</i> -Attribut; das Attribut der aufrufenden Shell wird übernommen (nur für Funktionen).
<code>-x</code>	markiert Namen für den Export.

Tabelle 27.9 »local«-Parameter

Mit der Deklaration können Sie nicht nur den Typ definieren, sondern auch Variablennamen innerhalb einer Funktion benutzen, die Sie bereits im Skript verwendet haben, ohne dass diese überschrieben werden.

Typdefinition von Variablen

Eine feste Definition des Variablentyps kann unter Umständen notwendig sein, etwa dann, wenn der Inhalt einer Datei in ein *Array* eingelesen werden soll. Das `declare`-Kommando ist dafür vorgesehen. Sie können damit vorab definieren, welchem Typ die Variable entsprechen soll. `declare` verarbeitet die gleichen Parameter wie `local`. Wenn Sie `declare` innerhalb einer Funktion verwenden, agiert es wie das `local`-Kommando und definiert den Gültigkeitsbereich der Variablen nur innerhalb dieser Funktion. Rufen Sie `declare` ohne Parameter auf, erhalten Sie eine Ausgabe aller deklarierten Variablen (inklusive des *Environments*).

Um eine Variable zu entladen, also zu löschen, verwenden Sie das `unset`-Kommando. Es löscht die Variable und deren Inhalt. Mit dem Parameter `-f` können Sie auch Funktionen löschen. Der Parameter `-v` erwartet eine Variable. Ohne Parameter wird davon ausgegangen, dass Sie eine Variable entladen wollen. Allerdings können Sie mit dem Befehl `unset` keine Konstanten (*readonly*-Variablen) löschen.

Funktionen

Selbstverständlich können Sie in Shell-Skripten auch Funktionen definieren und verwenden. Beachten Sie dabei aber, dass es sich bei Shell-Skripten um Programme handelt, die sequenziell abgearbeitet werden. Daher müssen Sie die Funktion erst definieren, bevor Sie sie verwenden können. Eine Funktion definieren Sie wie folgt:

```
[ function ] name () compound-command [redirection]
```

Listing 27.7 Syntax für eine Funktionsdefinition

Listing 27.7 erstellt eine Funktion mit dem Namen `name`. Die Einleitung einer Funktion mit dem Kommando `function` ist optional – aus Gründen der Lesbarkeit sollten Sie dieses aber verwenden. Der Bereich `compound-command` enthält die Befehle innerhalb der Funktion und wird von geschweiften Klammern umschlossen. Funktionen geben immer den Wert des letzten Kommandos zurück – falls ein anderer Wert zurückgegeben werden soll, müssen Sie das reservierte Wort `return` verwenden. Abschließend kann die Ausgabe der Funktion auch umgeleitet werden. Dies geschieht über die allgemeine Syntax mittels einer schließenden spitzen Klammer (`>`). Eine einfache Funktionsdefinition kann wie folgt aussehen:

```
function Ausgabe () {  
    echo "Ich bin eine Funktion"  
}
```

Listing 27.8 Funktionsdefinition

test-Kommando

Eine der Kernkomponenten bei der Programmierung sind Fallentscheidungen. Sie werden durch *if*-Abfragen realisiert. In Shell-Skripten werden diese mit eckigen Klammern umschlossen. Das interne `test`-Kommando wird über zwei eckige Klammern referenziert. Mit dem `test`-Kommando ist es möglich, viele Prüfungen direkt vorzunehmen, ohne weitere Programmaufrufe vornehmen zu müssen. Mit dem `test`-Kommando können Sie die Prüfungen vornehmen, die Sie in Tabelle 27.10 sehen.

Aufruf	Bedeutung
<code>expr</code>	gibt <i>true</i> zurück, wenn <code>expr</code> <i>true</i> ist.
<code>! expr</code>	gibt <i>true</i> zurück, wenn <code>expr</code> <i>false</i> ist.
<code>expr1 -a expr2</code>	gibt <i>true</i> zurück, wenn <code>expr1</code> und <code>expr2</code> <i>true</i> sind.
<code>expr1 -o expr2</code>	gibt <i>true</i> zurück, wenn <code>expr1</code> oder <code>expr2</code> <i>true</i> ist.
<code>-d file</code>	gibt <i>true</i> zurück, wenn <code>file</code> ein Verzeichnis ist.
<code>-e file</code>	gibt <i>true</i> zurück, wenn <code>file</code> existiert.
<code>-f file</code>	gibt <i>true</i> zurück, wenn <code>file</code> eine Datei ist.
<code>-s file</code>	gibt <i>true</i> zurück, wenn die Größe der Datei <code>file</code> größer als 0 ist.
<code>-x file</code>	gibt <i>true</i> zurück, wenn die Datei <code>file</code> ausführbar ist.
<code>file1 -nt file2</code>	gibt <i>true</i> zurück, wenn <code>file1</code> neuer ist als <code>file2</code> .
<code>file1 -ot file2</code>	gibt <i>true</i> zurück, wenn <code>file1</code> älter ist als <code>file2</code> .
<code>-z string</code>	gibt <i>true</i> zurück, wenn die Länge von <code>string</code> gleich 0 ist.
<code>-n string</code>	gibt <i>true</i> zurück, wenn die Länge von <code>string</code> nicht gleich 0 ist.

Tabelle 27.10 Auszug: Prüfungen des »test«-Kommandos

Die Shell kann auch Vergleiche mit regulären Ausdrücken (`=~`, siehe Listing 27.4) durchführen. Das kann das `test`-Kommando nicht.



27.3.4 Logging in Skripten

Wenn Shell-Skripte eine gewisse Größe bekommen, mehrere systemkritische Operationen ausführen oder sogar interaktiv von Benutzern gesteuert werden, ist es sinnvoll, ein eigenes Logfile anlegen zu lassen. Selbstverständlich können Sie aus einem Shell-Skript heraus mit

dem Programm `logger` Meldungen direkt an den `Syslog`-Daemon schicken. Oft ist es aber sinnvoller, ein eigenständiges Log zur Auswertung zu besitzen. Damit Sie nicht das Rad neu erfinden müssen, stellen wir Ihnen in diesem Abschnitt das Programm `log4bash` vor.

Mit dem von Philip Patterson entwickelte Pendant zu `log4j` (für Java) oder `log4perl` (für Perl) können Sie, eigene Logs Ihrer Shell-Skripte erstellen. Das Tool können Sie von der Webseite des Entwicklers herunterladen: <https://www.gossiplabs.org/log4bash.html>



Falls Sie keine `bash` verwenden, können Sie auch das »allgemeinere« `log4sh` verwenden. Dieses steht unter <https://github.com/kward/log4sh> zum Download bereit.

Damit Sie `log4bash` in Ihren Skripten verwenden können, müssen Sie es einbinden. Erstellen Sie ein neues Verzeichnis `scripts` in Ihrem Homeverzeichnis, und speichern Sie das entpackte `log4bash` dort. Erstellen Sie nun die Datei `myfirstlog.bash` mit folgendem Inhalt:

```
#!/bin/bash
source /home/dirk/scripts/log4bash.sh
log4bash 0 Meldung
```

Listing 27.9 Beispielskript »myfirstlog.bash«

Durch das Einbinden können Sie die Funktion `log4bash` direkt in Ihrem Skript aufrufen. Der Aufruf des Skripts erzeugt eine Log-Datei, die zum Skriptnamen passt, an die das Datum angehängt ist und die auf `.log` endet.

In unserem Beispiel heißt sie also `myfirstlog.bash.jjjjmmdd.log`, wobei `jjjjmmdd` für das heutige Datum steht. In dem neu erstellten Log findet sich folgende Meldung:

```
2018-10-01 19:09:49.748490329 CET|DEBUG|0|main|meldung
```

Listing 27.10 Inhalt von »myfirstlog.bash.jjjjmmdd.log«

Die Log-Einträge bestehen aus fünf Elementen:

1. Datum, Uhrzeit und Zeitzone

Nach dem Muster `YYYY-MM-DD HH:MM:SS.N Z` wird das Datum angegeben.

2. Log-Level

Das Log-Level wird über folgende Zahlenwerte angegeben:

- 0 = DEBUG
- 1 = INFO
- 2 = WARN
- 3 = ERROR
- 4 = CRITICAL
- 5 = FATAL

3. Zeilennummer

Ohne Angabe wird die Zeilennummer (wie im Beispiel) auf 0 gesetzt. Diese können Sie für weitere (eigene) Auswertungszwecke beliebig vergeben – zum Beispiel in der gleichen Notation wie HTTP-Statuscodes.

4. Funktionsname

Ohne Ihre ebenfalls beliebige Angabe wird als Funktionsname `main` gesetzt.

5. Meldung

Abschließend wird die übergebene Meldung aufgeführt.

Der Aufruf in Listing 27.11 würde als einfaches Beispiel eine Log-Meldung der Stufe `INFO` mit der Zeilennummer 20, dem Funktionsnamen `Funktion` und der Meldung `Meldung aus mehreren Wörtern` erstellen. Das Listing zeigt auch die Darstellung:

```
log4bash 1 "Meldung aus mehreren Wörtern" 20 Funktion
```

```
2018-10-01 19:29:50.627352792 CET|INFO|20|Funktion|Meldung aus mehreren Wörtern
```

Listing 27.11 Alternativer Aufruf von »`log4bash.sh`«

Damit lässt sich ein gut durchsuchbares Basis-Logging realisieren. Das Verhalten von `log4bash` lässt sich mit den Variablen aus Tabelle 27.11 beeinflussen.

Variable	Bedeutung
L4B_DELIM	definiert das Trennzeichen (Default:).
L4B_DATE	Datumskommando (Default: date)
L4B_DATEFORMAT	Datumsformat-Spezifikation (Default: +%Y-%m-%d %H:%M:%S.%N %Z)
L4B_VERBOSE	gibt an, ob die Log-Meldung auch auf dem Bildschirm ausgegeben werden soll (Default: false).
L4B_LABEL	Feld (Array) mit den Log-Levels
L4B_LOGFILE	Namensgebung der Log-Datei (Default: <SKRIPT_NAME>.<DATUM>.log)
L4B_DEBUGLVL	Schwellenwert, der angibt, ab wann Meldungen ins Log geschrieben werden (Default: 0 = Alles wird geloggt.)

Tabelle 27.11 Anpassungsmöglichkeiten: »`log4bash`«

Mit diesen Variablen können Sie die Log-Erzeugung mit `log4bash` Ihren Bedürfnissen anpassen. In Listing 27.12 werden die Variablen gesetzt, und es wird eine Besonderheit dargestellt:

```
#!/bin/bash
source log4bash.sh

declare -a L4B_LABEL=(DEBUG INFO WARN ERROR CRITICAL FATAL ENDOFWORLD)
L4B_DEBUGLVL=1
L4B_DELIM=" "
L4B_DATE="date"
L4B_DATEFORMAT="+%Y%m%d-%H%M%S.%N"
L4B_VERBOSE=true
L4B_LOGFILE="skript.log"

log4bash 1 "1. Meldung" 15 Hauptprogramm

L4B_VERBOSE=false

log4bash 0 "2. Meldung" 25 Hauptprogramm
log4bash 1 "3. Meldung" 25 Hauptprogramm
```

Listing 27.12 Angepasstes »log4bash«: »mysecondlog.bash«

Nach dem Ausführen des Skripts findet sich in der angegebenen Log-Datei `skript.log` folgender Inhalt:

```
20181001-135700.980323531 INFO 15 Hauptprogramm 1. Meldung
20181001-135701.026624750 INFO 25 Hauptprogramm 3. Meldung
```

Listing 27.13 »skript.log«

Bereits während der Ausführung wurde die erste Meldung auch auf dem Bildschirm ausgegeben. Meldung 3 hingegen findet sich nur im Log wieder, da die Variable `L4B_VERBOSE` vor der Verarbeitung auf `false` gesetzt wurde. Meldung 2 wurde weder auf dem Bildschirm ausgegeben noch in die Log-Datei geschrieben. Dies ist darauf zurückzuführen, dass der zweiten Meldung der Log-Level 0 zugewiesen wurde. Weil wir die Variable `L4B_DEBUGLVL` auf den Wert 1 gesetzt haben, wurde `log4bash` angewiesen, nur Log-Einträge zu protokollieren, die mindestens dem Log-Level 1 entsprechen.

27.4 Tipps und Tricks aus der Praxis

Oft stößt der ambitionierte Sysadmin an Grenzen, die er für unüberwindbar hält, oder er entwickelt unnötig viel Code, um eine Problemstellung zu lösen. In diesem Abschnitt zeigen wir Ihnen eine Auswahl von Tipps und Tricks, die Ihnen sicherlich in der einen oder anderen Situation helfen werden.

27.4.1 Aufräumkommando

Ein sauberer Arbeitsplatz gehört für jeden Handwerker zum guten Ton, so auch für den Sysadmin. In diesem Abschnitt geht es allerdings nicht um den Schreibtisch, sondern vielmehr um das Sauberhalten der Festplatte. Schnell laufen automatisiert erstellte Log-Dateien oder Reports über. Daher sollten Sie diese periodisch entfernen und nur die letzten (relevanten) Daten aufbewahren. Dafür eignet sich folgender Einzeiler hervorragend:

```
find $PFAD -type f -mtime +$ALTER -exec rm -f {} \+
```

Listing 27.14 Alte Daten entfernen

Dieser *find*-Befehl findet alle Dateien im Verzeichnis *\$PFAD* und löscht die Dateien, die älter als *\$ALTER* Tage sind. Selbstverständlich können Sie diesen Befehl auch so modifizieren, dass die alten Daten vorher gesichert werden. Nach dem Parameter *-exec* folgt der Befehl, den *find* ausführen soll, wenn eine Datei den Suchkriterien entspricht. Die jeweilige Datei wird dann anstatt der geschweiften Klammern eingesetzt. Alternativ können Sie auch folgende Variante verwenden, wenn es um das reine Löschen und nicht um erweiterte Funktionen geht:

```
find $PFAD -type f -mtime +$ALTER -delete
```

Listing 27.15 Alte Daten entfernen, Variante

27.4.2 IFS

Einer der großen Spielverderber ist der *Internal Field Separator (IFS)*. In dieser Variablen sind die Trennzeichen spezifiziert, anhand derer die Shell eine Trennung vornimmt. Diese Trennungen werden zum Beispiel vorgenommen, wenn Sie innerhalb einer *For*-Schleife eine Datei ausgeben lassen. Da zu den Trennzeichen Whitespaces und auch das Newline (*\n*) gehören, wird beim Durchlaufen der *For*-Schleife nicht jede Zeile der Variablen zugeordnet, sondern jedes Wort. Um dies zu vermeiden, können Sie den *IFS* umdefinieren. Erstellen Sie eine Textdatei *text.txt* mit mehreren Zeilen und beliebigem Inhalt. Erstellen Sie anschließend das Skript *ifs_changer.sh* mit folgendem Inhalt:

```
#!/bin/bash
IFS=%
for var in $(< text.txt); do
  echo $var
done
```

Listing 27.16 »ifs_changer.sh«

Dadurch, dass Sie den *IFS* auf den Wert *%* gesetzt haben, gibt das Skript den Inhalt der Datei *text.txt* zeilenweise aus. Entfernen Sie die Zeile *IFS=%*, so wird jedes Wort der Datei in einer eigenen Zeile ausgegeben.

27.4.3 Datumsmagie

Die Berechnung von Daten kann eine mathematische Herausforderung werden. Wenn es nur darum geht, einen festen Termin, zum Beispiel »vor zwei Tagen« oder »in drei Monaten«, zu ermitteln, muss zum Glück kein mathematisches Konstrukt aufgebaut werden.

Das Programm *date* kann mit dem Parameter *-d* Datumsangaben mithilfe relativer Angaben berechnen (siehe Listing 27.17).

```
dirk@example:/# date
Mon Jun  5 08:23:11 CEST 2023
```

```
dirk@example:/# date -d '2 days ago'
Mon Jun  5 08:23:11 CEST 2023
```

```
dirk@example:/# date -d '1 month 2 days'
Fri Jul  7 08:23:54 CEST 2023
```

```
dirk@example:/# date -d '1 years 2 month 3 days'
Thu Aug  8 08:24:12 CEST 2024
```

```
dirk@example:/# date -d '10 minutes 5 seconds'
Mon Jun  5 08:34:39 CEST 2023
```

Listing 27.17 Daten relativ berechnen: »date -d«

Durch die Angabe der Werte wird das Datum entweder für die Zukunft oder mit der Option *ago* für die Vergangenheit, vom Moment der Ausführung an, berechnet. Dabei versteht *date* die Begriffe *second(s)*, *minute(s)*, *hour(s)*, *day(s)*, *month(s)* und *year(s)*.

27.4.4 E-Mails aus einem Skript versenden

Über Ergebnisse aus Reports oder anderen Verarbeitungen möchten Sie sicherlich umgehend informiert werden. Dazu eignet sich der Versand der Daten in einer E-Mail. Dazu können Sie das Tool *mailx* verwenden. Falls dieses auf Ihrem System nicht vorhanden ist, installieren Sie das Paket *heirloom-mailx* oder *bsd-mailx*. Anschließend können Sie E-Mails aus einem Shell-Skript mit folgender Zeile versenden:

```
mailx RECIPIENT@example.net -s "Betreff Zeile" < EMAILTEXT.txt
```

Listing 27.18 Beispiel: E-Mail-Versand mit »mail«

Der Empfänger wird direkt (ohne Parameter) als Option angegeben. Die Betreffzeile wird über den Parameter *-s* angegeben.



E-Mail-Konfiguration

Für den Einsatz von `mailx` muss das System über eine E-Mail-Konfiguration (`postfix`, `exim` oder ein anderer MTA) verfügen!

27.4.5 Interaktive Programme steuern

Leider lassen sich nicht alle Programme vollständig über Parameter steuern. Meist ist dies nicht mehr möglich, wenn eine Anmeldung vorausgesetzt wird. Durch die sequenzielle Verarbeitung sind Shell-Skripte eigentlich nicht in der Lage, interaktive Programme zu steuern.

Dafür wurde das Programm `expect` geschaffen, das Rückgabewerte interpretieren und entsprechend reagieren kann. Der Einsatz von `expect` ist aber nicht immer notwendig. Kleinere Konstrukte kann die Shell auch selbst verarbeiten.

Das Skript aus Listing 27.19 verbindet sich mittels `telnet` über den TCP-Port 1234 mit dem `localhost`, setzt zwei Befehle ab und speichert die Ausgabe in der temporären Datei `/tmp/ausgabe.txt`:

```
#!/bin/bash
(
echo "get info"
sleep 3
echo "quit"
) | telnet 127.0.0.1 1234 > /tmp/ausgabe.txt
```

Listing 27.19 Steuerung interaktiver Programme

Durch die Klammerverarbeitung der Shell wird dem Programm `telnet` nur die Ausgabe der Befehle übergeben. Selbst das `sleep` wird befolgt, bevor die Verbindung über `quit` beendet wird.

Kapitel 28

Konfigurationsmanagement mit Ansible

Ansible ist eine Software zur Konfiguration und Administration von Computern. Die Kernaufgabe eines solchen Konfigurationsmanagementsystems ist es, eine definierte Zustandsbeschreibung eines Hosts umzusetzen. Klassische Tools wie Puppet setzen dazu auf den Zielsystemen installierte Agenten voraus, die dann für die Umsetzung zuständig sind. Ansible hingegen kommt komplett ohne Agenten aus, da es über SSH arbeitet. Auf den Zielsystemen werden im Wesentlichen lediglich ein SSH-Server und eine Python-Installation benötigt – auf Linux-Systemen völlig unproblematische Voraussetzungen.

28

28.1 Einführung und Installation

In diesem Abschnitt möchten wir Ihnen die grundlegenden Eigenschaften von Ansible vorstellen. Danach beschreiben wir unsere exemplarische Testumgebung und gehen dann so gleich auf die Ansible-Installation ein.

28.1.1 Was ist Ansible?

Ansible ist ein Werkzeug zur Lösung von Problemen, die sich meist auf der »Ops«-Seite der DevOps-Welt stellen. Jeder, der schon einmal für eine kleine Anzahl von Servern verantwortlich war, wird bestätigen können, dass eine rein manuelle Administration bzw. Konfiguration zumindest sehr fehleranfällig, aufwendig und schlecht nachvollziehbar ist. Und jeder, der schon einmal für eine große Anzahl von Servern verantwortlich war, wird bestätigen, dass die rein manuelle Administration im Prinzip unmöglich ist. Vielleicht haben Sie in einer solchen Situation schon mit (Shell-)Skripten gearbeitet, oder Sie sind sogar bereits mit einem Alternativprodukt wie Puppet, Chef, SaltStack oder CFEngine vertraut. Das Werkzeug Ansible bietet Ihnen u. a. folgende Möglichkeiten:

- ▶ Automatisierung der Provisionierung von Systemen
- ▶ Automatisierung des Software-Deployments
- ▶ Konfigurationsmanagement

Ansible ist in der Tat ein äußerst flexibles Tool; Sie können es benutzen, um auf einer Maschine eine Software auszurollen, oder Sie können damit Konfigurationen auf Hunderten von Servern kontrollieren. Zudem ist es (relativ) leicht zu erlernen, und den meisten Anwendern macht es sogar Spaß!

Grundmerkmale

Ansible zeichnet sich durch folgende Eigenschaften aus:

- ▶ Ansible ist in Python programmiert. Ursprünglich wurde es in Python 2 entwickelt, ab Ansible Version 2.5 wurde aber auch Python 3 vollständig unterstützt.
- ▶ Ansible verwaltet seine Hosts in den allermeisten Fällen über SSH und benötigt auf diesen lediglich Python (zuzüglich einiger Python-Module in speziellen Fällen). Insbesondere ist Ansible *agentenlos*, da es keine speziellen Dienste auf den Zielsystemen benötigt.
- ▶ Ansible nutzt hauptsächlich YAML als Konfigurationssprache.
- ▶ Für Ansible stehen Tausende von Modulen für die verschiedensten Verwaltungsaufgaben zur Verfügung.
- ▶ Ansible bietet recht schnell Erfolgserlebnisse, skaliert aber auch gut in komplexen Szenarien.
- ▶ Ansible ist sehr ordentlich dokumentiert (siehe <http://docs.ansible.com>).

Kernkomponenten

In der Regel findet man in einem Ansible-Projekt folgende Komponenten:

1. Das *Inventory* ist ein Verzeichnis der Maschinen, die mit Ansible administriert werden sollen. Dieses muss in der Regel vom Anwender erstellt werden.
2. Die *Playbooks* sind vom Benutzer zu erstellende Prozeduren, die in einzelnen Schritten (*Tasks*) beschreiben, wie die jeweiligen Konfigurationsziele zu erreichen sind.
3. Die *Module* sind von Ansible zur Verfügung gestellte Funktionseinheiten, die die eigentliche Arbeit erledigen. Jeder Schritt in einem Playbook ist letztlich nichts anderes als der Aufruf eines Moduls.
4. *Rollen* dienen dazu, größere Projekte modular, wartbar und wiederverwendbar zu gestalten. Mindestens eines haben sie mit den Klassen aus der objektorientierten Programmierung gemeinsam: Ganz zu Anfang braucht man sie nicht, und später will man sie nicht mehr missen.

Was bedeutet »Ansible«?

Der Name »Ansible« kommt aus der Science-Fiction. Ein *Ansible* ist dort ein Gerät zur überlichtschnellen Kommunikation. Ursprünglich von Ursula K. Le Guin ersonnen, haben viele andere Autoren die Idee adaptiert.

Ganz konkret hat der Ansible-Erfinder Michael DeHaan den Begriff aus dem Buch »Ender's Game« von Orson Scott Card entliehen. Dort wird das Ansible ebenfalls benutzt, um über sehr große Entfernungen mit Flotten oder Stützpunkten zu kommunizieren.

28.1.2 Geschichte und Versionen

Im Jahr 2012 stellte Michael DeHaan die erste Version von Ansible vor. Das von ihm gegründete Unternehmen *Ansible Inc.* wurde Ende 2015 von Red Hat übernommen. Seitdem wird die Weiterentwicklung maßgeblich von Red Hat beeinflusst, was dem Produkt bislang sehr gut tut. Mit bzw. nach der Ansible-Version 2.10 wurden das Versionierungsschema und vor allem die interne Organisation der Weiterentwicklung stark verändert. Seitdem werden der Kern von Ansible und die Zusatzkomponenten wie Module und Plug-ins unabhängig voneinander entwickelt. Der Ansible-Kern wird mit dem alten Versionierungsschema weiterentwickelt und heißt nun *Ansible-Base* (Version 2.10) bzw. *Ansible-Core* (Version ≥ 2.11). Die in großer Zahl existierenden Zusatzkomponenten heißen nun *Collections* und folgen jeweils einem individuellen Versionierungsschema.

Für die Endanwender am interessantesten dürfte nun das *Ansible Community Package* sein, beginnend mit Version 3.0. Dies ist eigentlich nur eine Meta-Distribution, die die Version des enthaltenen Ansible-Kerns und die Auswahl und Versionen der mitgelieferten *Collections* festlegt. Die Versionsnummern folgen nun außerdem dem Prinzip der semantischen Versionierung (*Major.Minor.Patch*). In der Übersicht stellt sich das wie folgt dar:

- ▶ Ansible 2.8 oder älter: klassisches Komplettpaket (für uns: »Steinzeit«)
- ▶ Ansible 2.9: klassisches Komplettpaket, *Collections* als Tech Preview
- ▶ Ansible 2.10: Abschluss der Vorbereitungen, Aufteilung in *Ansible-Base* und *Collections*
- ▶ Ansible 3.0: *Community Package*, bestehend aus *Ansible-Base* 2.10 + *Collections*
- ▶ Ansible 4.0: *Community Package*, bestehend aus *Ansible-Core* 2.11 + *Collections*
- ▶ Ansible 5.0: *Community Package*, bestehend aus *Ansible-Core* 2.12 + *Collections*
- ▶ Ansible 6.0: *Community Package*, bestehend aus *Ansible-Core* 2.13 + *Collections*
- ▶ Ansible 7.0: *Community Package*, bestehend aus *Ansible-Core* 2.14 + *Collections*

Grundlage für unsere Betrachtungen soll eine Ansible-Version ≥ 2.10 sein; falls es angebracht ist, gehen wir punktuell aber auch kurz auf Unterschiede oder Inkompatibilitäten zu Vorgänger- bzw. Nachfolgerversionen ein.

28.1.3 Setup/Laborumgebung

Wenn Sie mit Ansible arbeiten möchten, benötigen Sie zunächst einmal irgendeinen UNIX-artigen Host, auf dem die Ansible-Software installiert wird (den sogenannten *Control Host* oder auch die *Control Machine*). Jede gängige halbwegs aktuelle Linux-Distribution ist dazu geeignet, aber es ginge auch mit macOS, Solaris etc.

Außerdem benötigen Sie noch mindestens einen zusätzlichen Host, der via Ansible administriert werden soll (einen sogenannten *Target Host* oder *Managed Node*). Wir setzen hier ebenfalls Linux-Systeme voraus; grundsätzlich können Sie mit Ansible aber auch Windows-

Systeme oder Netzwerk-Devices administrieren. Die wichtigste Voraussetzung an dieser Stelle ist, dass der Control Host seine Target Hosts (im Falle von Linux) über SSH erreichen kann. Vieles, das wir in der Folge besprechen, wird sich auf die Laborumgebung aus Abbildung 28.1 beziehen, die Sie aber keinesfalls genau so nachbauen müssen.

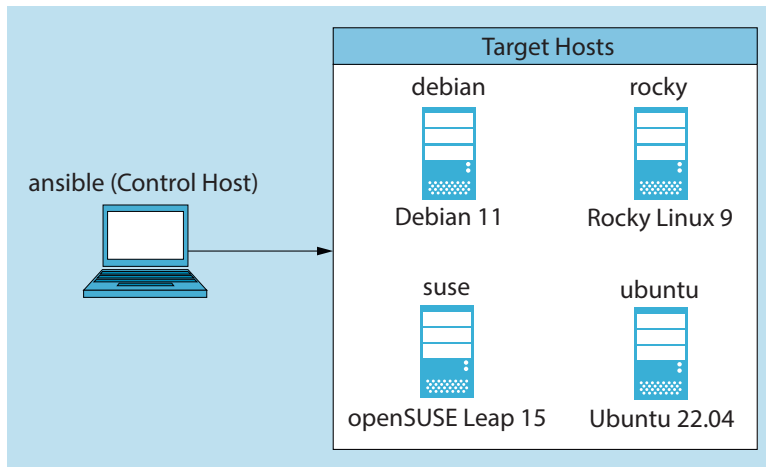


Abbildung 28.1 Unser Beispiel-Setup

Im Labor des Verfassers dieser Zeilen wird dieses Szenario mit *Vagrant* provisioniert, deswegen taucht später mitunter ein User *vagrant* auf, der uneingeschränkte *sudo*-Rechte hat. Wie gesagt: Bei Ihnen darf das gerne völlig anders aussehen.

In besagtem Labor ist der Control Host übrigens ein Debian-System, weswegen an wenigen Stellen mitunter ein `apt install`-Kommando präsentiert wird. Bei Ihnen könnte das dann ein `dnf install` oder `zypper install` sein – je nach eingesetzter Linux-Variante.

28.1.4 Ansible-Installation auf dem Control Host

Ansible ist in den Paketquellen aller wichtigen Distributionen vorhanden. Tabelle 28.1 zeigt den Stand der Dinge im Frühjahr 2023.

Distribution	paketierte Version
Debian 11	2.10.8
Rocky Linux 9	2.13.3 (nur <i>ansible-core</i>)

Tabelle 28.1 Ansible-Versionen in verschiedenen Distributionen

Distribution	paketierte Version
CentOS Stream 9	2.14.2 (nur <i>ansible-core</i>)
openSUSE Leap 15.4	2.9.27
Ubuntu 22.04 LTS	2.10.8

Tabelle 28.1 Ansible-Versionen in verschiedenen Distributionen (Forts.)

Anmerkung

Der Deutlichkeit halber möchten wir hier noch mal klarstellen: Da Ansible *agentenlos* ist, ist die Installation der Ansible-Software *nur* auf dem Control Host nötig.

Auf den Zielsystemen benötigen Sie momentan nichts (außer SSH und Python).



Die Installation aus Distributionspaketen ist im Allgemeinen recht schnell erledigt:

► Debian 11

```
# apt update
# apt install ansible
```

► Rocky Linux 9 / CentOS Stream 9

Wie in Tabelle 28.1 ersichtlich, ist im Standardlieferungsumfang von Rocky Linux 9 / CentOS Stream 9 nur das *ansible-core*-Paket vorhanden. Sobald Sie Module benötigen, die nicht in diesem Basispaket enthalten sind, müssten Sie diese direkt aus dem Internet nachinstallieren (siehe Abschnitt 28.7.1). In vielen Umgebungen wird die Verwendung des *ansible*-Packages aus dem EPEL-Zusatzrepository wohl einfacher sein, weswegen wir hier erst einmal diesen Weg empfehlen:

Zunächst aktivieren Sie das EPEL-Zusatzrepository:

```
# dnf install epel-release
# dnf install ansible
```

► openSUSE Leap 15

```
# zypper install ansible
```

Als diese Zeilen entstanden sind, gab es kein halbwegs offizielles Zusatzrepo, aus dem man eine aktuellere Version hätte beziehen können. Sie müssten im Bedarfsfall also auf eine Installation mit PIP (siehe unten) ausweichen.

► Ubuntu 22.04 LTS

```
$ sudo apt update
$ sudo apt install ansible
```

Wenn die Installation erfolgreich beendet wurde, machen Sie doch gleich einen Testaufruf:

```
$ ansible --version
ansible 2.10.8
  config file = None
  [...]
```

Lassen Sie uns auf dem Control Host auch noch `/etc/hosts`-Einträge für alle beteiligten Hosts vornehmen, denn auch im Testlabor wollen wir das Arbeiten mit IP-Adressen möglichst vermeiden. Wenn Ihnen in der Realität eine DNS-basierte Namensauflösung zur Verfügung steht, können Sie sich diesen Schritt natürlich sparen:

```
# Bitte entsprechend Ihrem Netzwerk anpassen:
192.168.150.10  debian.example.org  debian
192.168.150.20  rocky.example.org   rocky
192.168.150.30  suse.example.org    suse
192.168.150.40  ubuntu.example.org  ubuntu
```

Listing 28.1 »`/etc/hosts`«: exemplarischer Ausschnitt

Installation via PIP (+ Virtualenv)

Sollte Ihre Distribution nur ein veraltetes bzw. gar kein geeignetes Paket bereitstellen und sollten Zusatzrepositories für Sie keine Option sein oder möchten Sie gar mehrere Ansible-Versionen parallel betreiben, dann gibt es als zweite Möglichkeit die Installation über das Python-Paketmanagement.



Anmerkung

Wenn das Ansible-Paket Ihrer Distribution bzw. Ihres Zusatzrepositories hinreichend aktuell ist und Sie sich auch nicht für das Virtualenv-Feature von Python interessieren, können Sie gern den Rest dieses Abschnitts überspringen und gleich in Abschnitt 28.1.5 weiterlesen.

Zunächst müssen Sie als Root für die nötigen Voraussetzungen sorgen. Sie benötigen zwei Distributionspakete:

```
# apt install python3-pip python3-venv
```

Unter SUSE und Rocky bräuchten Sie nur `python3-pip`, denn `venv` ist ein Standardmodul, das Python 3 schon an Bord hat. (Fragen Sie nicht, warum sich das in der Debian-Welt noch nicht herumgesprochen hat.)

Der schnellste Weg zur Ansible-Installation wäre nun die direkte Installation über das Python-Paketmanagement, wofür Sie auch nur das `python3-pip`-Paket gebraucht hätten:

Als Root, wenn Sie das wirklich wollen:

```
# pip3 install ansible
```

Eine gute Alternative ist die Installation in einer Virtualenv-Umgebung. Diese Methode hat den Vorteil, dass Sie als unprivilegierter Benutzer arbeiten können, und sogar beliebig viele Versionen parallel betreiben könnten. Und Sie vermeiden damit das potenzielle Problem, dass PIP irgendwelche Änderungen am System durchführt, die nur schwer wieder rückgängig zu machen sind. Geben Sie die folgenden Befehle also als unprivilegierter User ein:

Eine virtuelle Umgebung einrichten (das Verzeichnis ist frei wählbar):

```
$ python3 -m venv ~/venv/ansible
```

Betreten der Umgebung:

```
$ source ~/venv/ansible/bin/activate
```

*Bei der *Erstnutzung* der Umgebung ggf. PIP aktualisieren.*

Zumindest, falls ansonsten in der Folge ernste Probleme auftreten:

```
$ pip3 install --upgrade pip
```

Bei Bedarf verfügbare Ansible-Versionen anzeigen.

Das ist gruselig und je nach PIP-Version doch wieder anders:

```
$ pip3 install --use-deprecated=legacy-resolver ansible==
```

```
[...]
```

Ansible-Installation

(ohne "=="-Version erhalten Sie die aktuellste stabile Version):

```
$ pip3 install ansible==3.4.0
```

```
[...]
```

```
$ ansible --version
```

```
ansible 2.10.17
```

```
[...]
```

Verlassen der Umgebung:

```
$ deactivate
```

Wenn Sie möchten, könnten Sie nun beliebig viele solcher Virtualenv-Umgebungen mit verschiedenen Ansible-Versionen verwalten.

28.1.5 Authentifizierung und Autorisierung auf den Target Hosts

Kommen wir nun zu den Target Hosts. Wir wollen auf ihnen mit Ansible administrative Tätigkeiten verrichten, also müssen pro Zielhost einige Fragen geklärt werden:

1. Welcher Benutzeraccount kann von außen via SSH erreicht werden, und welche Authentifizierungsmethode soll verwendet werden? Am gängigsten ist hier die Public-Key-Authentifizierung, aber mitunter stehen auch nur klassische Passwörter zur Verfügung.

Seltener anzutreffen, aber nicht unüblich ist noch die SSH-Kerberos-Authentifizierung. Diese einzurichten ist ein völlig eigenständiges und hochkomplexes Thema, das wir hier nicht behandeln können. Wenn Sie aber eine fertig eingerichtete Umgebung vor sich haben, ist aus Sicht der Ansible-Kommandozeile keine weitere Konfiguration erforderlich – ein gültiges Kerberos-Ticket kann von Ansible genauso verwendet werden wie von einem menschlichen User.

2. Wenn Ansible sich erfolgreich authentifiziert hat: Stehen dann bereits Root-Rechte zur Verfügung, oder muss noch eine Identitätsänderung bzw. Rechteerhöhung durchgeführt werden? Wenn ja, mit welcher Methode (su/sudo)?

Unsere hypothetische Laborumgebung ist diesbezüglich sehr einheitlich: Wir erreichen alle Systeme über den Account `vagrant` mit dem Passwort `vagrant`. Die Rechteerhöhung wird mit `sudo` durchgeführt.

28.1.6 Einrichten der SSH-Public-Key-Authentifizierung

Für den Betrieb von Ansible richten Sie typischerweise eine SSH-Public-Key-Authentifizierung vom Control Host zu den Target Hosts ein. Wir nutzen als Ausgangsaccount auf unserem Control Host einen unprivilegierten User; aus bereits erwähnten Gründen heißt dieser hier `vagrant`. Dauerhaftes Arbeiten als Root wäre auf dem Control Host jedenfalls weder nötig noch sinnvoll. Wir gehen davon aus, dass Sie wissen, was zur Einrichtung der Public-Key-Authentifizierung zu tun ist. Falls nicht, folgt hier eine Quick-and-dirty(!)-Vorgehensweise für die angenommene Testumgebung:

```
vagrant@ansible:~$ ssh-keygen  
(... alle Defaults akzeptieren ...)
```

Verteilen des Schlüssels:

```
vagrant@ansible:~$ ssh-copy-id vagrant@debian  
vagrant@ansible:~$ ssh-copy-id vagrant@rocky  
vagrant@ansible:~$ ssh-copy-id vagrant@suse  
vagrant@ansible:~$ ssh-copy-id vagrant@ubuntu
```



Anmerkung

Sie können natürlich auch gern einen Schlüssel mit einem vom Default abweichenden Dateinamen generieren bzw. verwenden (z. B. `~/ssh/ansible_key`). Bei `ssh-copy-id` müssen Sie diesen dann mittels `-i` angeben.

28.1.7 Ein Ad-hoc-Test ohne jegliche Konfiguration

Mit einem Ansible-ping-Aufruf können (und sollten!) Sie nun testen, ob die Ansible-Arbeitsumgebung so weit korrekt eingerichtet ist. Bevor es losgeht, noch zwei Anmerkungen:

- ▶ Wenn der Dateiname Ihres privaten SSH-Schlüssels vom Default abweicht, ergänzen Sie bitte jeweils `--private-key <PFAD>`.
- ▶ Die Angabe von `-u <USERNAME>` ist überflüssig, falls der Zielaccount genau wie der Ausgangsaccount lautet.

```
$ ansible all -i debian,rocky,suse,ubuntu -u vagrant -m ping
```

```
debian | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

```
[WARNING]: Platform linux on host rocky is using the discovered
Python interpreter at /usr/bin/python, but future installation
of another Python interpreter could change the meaning of that path.
See https://docs.ansible.com/[...] for more information.
```

```
rocky | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
[...]
```

Trotz aller eventuellen Warn- bzw. Hinweismeldungen sollten Sie hoffentlich feststellen können, dass alle Zielsysteme mit einem freundlichen `pong` antworten. Ab Version 2.8 ist Ansible wesentlich smarter geworden, was die Erkennung des Python-Interpreters auf der Gegenseite betrifft. Es lässt nun aber auch alle Welt mit ausgiebigen Warn- und Hinweismeldungen an seiner Klugheit teilhaben; wir werden das in Kürze per Konfiguration etwas ruhiger gestalten.

Sollten Sie an dieser Stelle ernsthafte Fehlermeldungen sehen, so haben diese in aller Regel mit einem tatsächlich nicht vorhandenen Python-Interpreter zu tun. Oder Ihr Ansible ist zu alt und erkennt die gegebenenfalls vorhandene Python-Version nicht richtig. Statten Sie im ersten Fall Ihre Zielsysteme mit einer Python-Installation aus, und im zweiten Fall raten wir Ihnen dazu, ein Ansible in einer hinreichend aktuellen Version zu betreiben.

Python 2, Python 3 oder Platform-Python?

Falls Sie ein minimales Rocky/RHEL/CentOS-8-Zielsystem mit am Start haben, dann erkennt Ihr Ansible möglicherweise dort den Python-Interpreter `/usr/libexec/platform-python`. Dabei handelt es sich um eine minimale Python-Variante, die gerade genug Funktionalität für Sys-

temwerkzeuge wie `dnf` liefert. Um hier eventuellen Problemen schon jetzt aus dem Weg zu gehen (und aus Gründen der Einheitlichkeit), sollten Sie dieses Zielsystem noch mit einem richtigen Python ausstatten:

Wahrscheinlich nur für ein RH-8-System erforderlich:

```
$ ssh rocky
[vagrant@rocky ~]$ su -
[root@rocky ~]# dnf install python3
[...]
Complete!
```



Anmerkung

Stören Sie sich nicht daran, dass Ansible danach immer noch den `platform-python`-Interpreter erkennt. Wir werden es im nächsten Abschnitt mit der Nase auf den »richtigen« Interpreter stoßen.

Ansonsten lässt sich zur Versionsproblematik von Python beruhigend feststellen, dass Ansible mit Python 2 genauso klarkommt wie mit Python 3. Die Python-Versionen auf dem Control Host und den Target Hosts müssen natürlich auch *nicht* übereinstimmen; man kann das Ganze also relativ entspannt sehen.

28.2 Basiseinrichtung und erstes Inventory-Management

In diesem Abschnitt finden Sie einen Vorschlag, wie Sie Ihre Arbeit mit Ansible auf Verzeichnisebene strukturieren können. Außerdem beschäftigen wir uns mit dem *Inventory* (sozusagen dem Verzeichnis der Target Hosts), das Sie für nahezu jedes Betriebsszenario von Ansible benötigen.

28.2.1 Verzeichnisstruktur einrichten

Ansible macht uns im Prinzip kaum feste Vorgaben, welche Strukturen wir auf Verzeichnis- und Dateiebene anzulegen haben. In aller Regel ist es sinnvoll, wenn Sie dabei Folgendes berücksichtigen:

- ▶ Ein Ansible-Projekt sollte sich innerhalb bzw. unterhalb eines einzigen Ordners befinden. Vorteil: Diesen Ordner können Sie jederzeit einfach sichern oder in ein Versionskontrollsystem einchecken, ohne dass Sie befürchten müssen, etwas Wichtiges vergessen zu haben.
- ▶ Wir wollen auf dem Control Host als unprivilegierter User arbeiten (wie schon beim Einrichten der SSH-Public-Key-Authentifizierung praktiziert).

Wegen des zweiten Punktes scheidet der Ordner */etc/ansible*, der zumindest früher bei der Installation aus Distributionspaketen standardmäßig angelegt wurde, schon einmal aus. Ignorieren Sie ihn einfach, falls er auf Ihrem System noch vorhanden sein sollte.

Jedes typische Ansible-Projekt braucht eine Konfiguration, ein Inventory und ein oder mehrere Playbooks. Starten wir mal ganz neutral mit einem Projektverzeichnis namens *~/ansible/projects/start*, das dann in Kürze all diese Dinge enthalten wird:

```
$ mkdir -p ~/ansible/projects/start && cd $_
```

Braucht man aber gleich schon eine solch tiefe Verzeichnishierarchie? Am Anfang im Prinzip nicht, ein Verzeichnis *hallo* oder *ansible-test* würde es genauso tun. Mit der vorgeschlagenen Struktur sind Sie aber schon ein wenig auf die Zukunft vorbereitet: Irgendwann werden Sie mehrere Projekte parallel managen wollen, und Sie werden nicht nur an Projekten arbeiten, sondern auch Rollen, Module, Plug-ins oder Collections entwickeln.

28.2.2 Grundkonfiguration (»ansible.cfg«)

Da wir Ansible im Folgenden mit einigen vom Default abweichenden Einstellungen betreiben wollen, benötigen wir eine Konfigurationsdatei.

Diese Datei wird an folgenden Stellen gesucht (der Reihe nach, die erste gewinnt):

1. Inhalt der Umgebungsvariablen `ANSIBLE_CONFIG`
2. *ansible.cfg* im aktuellen Verzeichnis
3. *~/ansible.cfg*
4. */etc/ansible/ansible.cfg*

Gemäß unseren Vorgaben bietet sich Möglichkeit Nr. 2 an. Legen Sie diese erste Version in Ihr Projektverzeichnis *~/ansible/projects/start/*:

```
[defaults]
inventory = ./inventory
```

Listing 28.2 »ansible.cfg«: erste Version einer Konfigurationsdatei

Ansible wird an dieser Stelle mit einem einfachen INI-Format konfiguriert. Die Erklärung der bislang einzigen Einstellung ist einfach: Sie setzen damit den Pfad zu einer (noch zu erstellenden) Inventory-Datei. Der Default wäre hier */etc/ansible/hosts* gewesen, was für uns völlig ungeeignet ist. Solange Sie sich nun im Basisordner des Projekts befinden, sollte Ansible die neue Konfigurationsdatei finden und nutzen:

```
$ ansible --version
ansible 2.10.8
  config file = /home/vagrant/ansible/projects/start/ansible.cfg
  [...]

```

**Achtung**

Ab jetzt ist etwas Disziplin gefragt, denn sobald Sie Ihr aktuelles Arbeitsverzeichnis ändern, würde Ansible sofort auf eine andere (oder gar keine) Konfiguration zurückgreifen, was nicht in unserem Sinne ist. Bitte achten Sie ab sofort also darauf, alle Ansible-Aufrufe stets im Basisordner Ihres Projekts zu tätigen!

Alternativ könnten Sie natürlich die oben erwähnte Umgebungsvariable `ANSIBLE_CONFIG` setzen. Sollten Sie das aber dauerhaft tun (z. B. via `.bashrc`), kann es sehr schnell Verwirrung geben, wenn Sie parallel mit mehreren Projekten arbeiten.

28.2.3 Erstellen und Verwalten eines statischen Inventorys

Um Ansible mitzuteilen, um welche Hosts es sich in Zukunft kümmern soll, können Sie im einfachsten Fall mit einer *Inventory-Datei* arbeiten (auch als *statisches Inventory* bezeichnet). Unser erstes Ziel soll sein, mittels einer solchen Datei den beliebigen und wichtigen Ping-Test zu ermöglichen:

```
$ ansible all -m ping
```

Probieren Sie es gern jetzt schon aus; es kann natürlich nicht funktionieren, da Ansible keine Idee hat, welche Hosts mit `all` gemeint sind. In unserer `ansible.cfg` haben wir ja schon den Pfad zur Inventory-Datei festgelegt (`./inventory`). Andere Dateinamen, die man oft an dieser Stelle sieht, sind `hosts` oder `hosts.ini`. Es folgt nun ein exemplarischer Inhalt für die Inventory-Datei, abgestimmt auf unsere Testumgebung:

```
[test_hosts]
debian
rocky
suse
ubuntu

[test_hosts:vars]
ansible_python_interpreter=/usr/bin/python3
ansible_ssh_common_args='-o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null'
ansible_user=vagrant
```

Listing 28.3 »inventory«

Die verwendete Syntax ist nicht ganz klar zu benennen; sagen wir einfach einmal, dass es sich um ein INI-ähnliches Format handelt. Aber davon abgesehen, sollte der Inhalt eigentlich recht gut verständlich sein:

- ▶ Mit Namen in eckigen Klammern können Sie Gruppen definieren (in diesem Beispiel: `test_hosts`). Es besteht keine Pflicht, Hosts in Gruppen zu organisieren, aber es ist sehr

üblich und nützlich. Eine spezielle Gruppe namens `all`, die alle Hosts enthält, ist übrigens stets automatisch vorhanden und muss nicht eigens von Ihnen definiert werden.

- ▶ Einzelhosts können mit zusätzlichen »Schlüssel=Wert«-Zuweisungen parametrisiert werden. Wir nutzen das in diesem Beispiel nicht.
- ▶ Hostgruppen können mit dem Zusatz `:vars` ebenfalls parametrisiert werden (hier: `test_hosts:vars`). Als exemplarische Anwendung haben wir Ansible durchgängig die Verwendung von Python 3 vorgeschrieben, und für alle Hosts unserer Testgruppe haben wir das Prüfen und Ablegen von SSH-Host-Keys deaktiviert, da Labor- bzw. Test-Hosts dazu neigen, öfter mal ihre Host-Keys zu wechseln (z. B. durch erneutes Provisionieren/Installieren). In unserer Testumgebung können wir auf Ereignisse wie

```
The authenticity of host '[...]' can't be established.
ECDSA key fingerprint is 8a:a3:a5:43:c6:e4:6c:11:3a:c9:2b:97:94:23:40:c9.
Are you sure you want to continue connecting (yes/no)?
```

oder

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
```

getrost verzichten; im Produktivbetrieb raten wir natürlich strengstens von dieser Einstellung ab (bzw. sollte sie dort nur verwendet werden, wenn Sie sich über die Implikationen völlig im Klaren sind)! Außerdem haben wir sicherheitshalber mit `ansible_user` den Account definiert, mit dem sich Ansible anmelden soll. (Defaultmäßig wird – wie auch bei SSH üblich – der lokale Aufrufaccount angenommen.)

- ▶ Sollten Sie übrigens in Ihrer Umgebung mit der untypischen UNIX-Passwort-Authentifizierung arbeiten, müssten Sie die entsprechenden Inventory-Hosts noch mit dem zusätzlichen Parameter `ansible_password=<LOGIN_PASSWORT>` versehen und zusätzlich das Paket `sshpass` installieren.

Machen Sie nun den Ping-Test – Sie sollten eine komplett grüne Ausgabe zu Gesicht bekommen:

```
$ ansible all -m ping
debian | SUCCESS => {
  "changed": false,
  "ping": "pong"
}
[...]
```

Die Reihenfolge der Server-Ausgaben ist hier nicht beeinflussbar und wegen paralleler Abarbeitung gewissermaßen zufällig.

28.2.4 Inventory-Aliasse und Namensbereiche

Bislang sind wir davon ausgegangen, dass die Hosts im Inventory über ihre dort verzeichneten Namen erreichbar sind; d. h., ein `server57.example.org` im Inventory wäre auch für `ssh`, `ping` und viele andere Tools unter diesem Namen erreichbar.

Das muss aber nicht der Fall sein. Vielleicht haben Ihre Target Hosts keine zugeordneten Namen oder es gibt zwar welche, aber sie gefallen Ihnen einfach nicht. Das können Sie alles mit *Aliassen* (also alternativen Namen) im Inventory ausgleichen; beispielsweise so:

```
s1 ansible_host=192.168.150.10
s2 ansible_host=192.168.150.20
s3 ansible_host=192.168.150.30
s4 ansible_host=192.168.150.40
```

Listing 28.4 »inventory«: Aliasse für IP-Adressen

oder so:

```
s1 ansible_host=debian.example.org
s2 ansible_host=rocky.example.org
s3 ansible_host=suse.example.org
s4 ansible_host=ubuntu.example.org
```

Listing 28.5 »inventory«: Aliasse für Namen

Dabei müssen die `ansible_host`-Namen natürlich wieder auflösbar sein. Für den Fall, dass Ihre Hosts nicht auf dem Standard-Port 22 erreichbar sind, steht Ihnen natürlich auch der Parameter `ansible_port` zur Verfügung:

```
s1 ansible_host=192.168.150.10 ansible_port=2201
s2 ansible_host=192.168.150.20 ansible_port=2202
s3 ansible_host=192.168.150.30 ansible_port=2203
s4 ansible_host=192.168.150.40 ansible_port=2204
```

Listing 28.6 »inventory«: Aliasse mit zusätzlicher Port-Angabe

Es gibt durchaus noch mehr solcher `ansible_*`-Parameter, aber für den Moment sollten Sie gut versorgt sein. Wer Interesse hat, kann gern einen Blick auf die Webseite http://docs.ansible.com/ansible/latest/user_guide/intro_inventory.html, Abschnitt »Connecting to hosts: behavioral inventory parameters«, werfen.

Namensbereiche

Wenn die Namen Ihrer Zielsysteme nach einem abzählbaren Schema vergeben sind, müssen Sie nicht jedes System einzeln auflisten. Sie können viel einfacher mit einer Range-Angabe arbeiten:

```
[webservers]
www[01:50].example.org
```

Listing 28.7 »inventory«: Bereiche von Hosts

Im Bereich der einstelligen Zahlen bekommen Sie dadurch Namen mit führenden Nullen, wie z.B. `www07.example.org`. Lassen Sie hingegen die führende Null in der Range-Angabe weg, bekommen Sie auch keine führenden Nullen im Ergebnis. Ranges von Buchstaben (z.B. `[a:h]`) funktionieren übrigens auch.

28.2.5 Jenseits von Ping

Ein funktionierender Ping-Test ist ein wichtiger Schritt auf dem Weg zur Basiseinrichtung, aber da er auf den Zielsystemen keinerlei Admin-Rechte erfordert, kann er leider *nicht* zeigen, ob auch die für alle ernsthafteren Aufgaben erforderliche Rechteerhöhung funktioniert.

Und natürlich tut sie das momentan nicht. Lassen Sie sich als Beweis einmal die erste Zeile der Datei `/etc/shadow` auf allen Zielsystemen ausgeben; eine Anforderung, für die Sie zweifellos Root-Rechte benötigen:

```
$ ansible all -a "head -1 /etc/shadow"
debian | FAILED | rc=1 >>
head: cannot open '/etc/shadow' for reading: Permission denied
[...]
```

Damit dies funktioniert, muss Ansible nach der Authentifizierung noch eine Rechteerhöhung durchführen. In unserer Laborumgebung hat der User `vagrant` uneingeschränkte, passwortfreie `sudo`-Rechte; wir kommen deswegen mit einem einzigen zusätzlichen Parameter (`ansible_become=yes`) zum Ziel:

```
[test_hosts]
debian
rocky
suse
ubuntu

[test_hosts:vars]
ansible_python_interpreter=/usr/bin/python3
ansible_ssh_common_args='-o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null'
ansible_user=vagrant
ansible_become=yes
```

Listing 28.8 »inventory«

Noch mal der Deutlichkeit halber: Das ist in unserer angenommenen Laborumgebung nur deswegen bereits ausreichend, weil der User `vagrant` mit passwortfreien `sudo`-Rechten aus-

gestattet ist. In der Folge erfahren Sie, welche Möglichkeiten Ihnen zur Verfügung stehen, wenn die Rechteerhöhung in Ihrer Umgebung anders organisiert ist.

Parameter zur Rechteerhöhung: Übersicht

Ganz allgemein stehen Ihnen zur Konfiguration der Rechteerhöhung folgende Parameter zur Verfügung:

- ▶ **ansible_become**
Der Hinweis an Ansible, dass auf dem Zielhost eine Rechteerhöhung stattfinden muss. Der Default ist `no`.
- ▶ **ansible_become_method**
Die dazu zu verwendende Methode. Der Default ist `sudo`.
- ▶ **ansible_become_user**
Der Zielaccount des Identitätswechsels. Der Default ist `root`, was in einem solchen Kontext allermeistens passt. Deswegen taucht dieser Parameter in obigem Beispiel auch überhaupt nicht auf.
- ▶ **ansible_become_pass**
Das zur Rechteerhöhung gegebenenfalls benötigte Passwort.

Zum Passwort gibt es natürlich noch etwas zu bemerken: Im Testbetrieb geht es ja möglicherweise in Ordnung, Klartextpasswörter in einer Konfigurationsdatei vorzuhalten; im Produktivbetrieb wahrscheinlich eher nicht. Welche Alternativen hätten Sie hier?

- ▶ Sie könnten sich stattdessen mit der Option `--ask-become-pass / -K` bei jedem Aufruf eines Ansible-Kommandos prompten lassen.
- ▶ Sie könnten zumindest im Falle von `sudo` per Konfiguration auf den Target Hosts auf die Passworteingabe verzichten; z.B. für alle Mitglieder der Gruppe `sudo`:

```
# Allow members of group sudo to execute any command
%sudo  ALL=(ALL:ALL) NOPASSWD: ALL
```

Listing 28.9 »/etc/sudoers«: Verzicht auf Passwort-Authentifizierung

Oder nur für einen speziellen Account, so wie es typischerweise auf Vagrant-Maschinen vorkonfiguriert ist:

```
vagrant ALL=(ALL) NOPASSWD: ALL
```

Listing 28.10 »/etc/sudoers«: Verzicht auf Passwort-Authentifizierung

Beide Optionen wollen vielleicht nicht so recht überzeugen, und für `su` hätten Sie immer noch gar keine automatische Lösung. Ich kann Sie letztlich an dieser Stelle nur auf später vertrösten, denn wenn Sie bald mehr über Ansible wissen, können Sie Hosts oder Gruppen durchaus auch an anderen Stellen parametrisieren und diese Stellen bei Bedarf auch

verschlüsseln. Informationen dazu finden Sie in Abschnitt 28.8. In unserem Testlabor sind wir mit dem aktuellen Inventory jedenfalls erst einmal für alle weiteren (administrativen) Aktionen gut aufgestellt:

```
$ ansible all -a "head -1 /etc/shadow"
debian | CHANGED | rc=0 >>
root:!:18333:0:99999:7:::
[...]
```

28.2.6 Ein etwas komplexeres Beispiel

Unsere Laborumgebung ist zugegebenermaßen ziemlich glatt geschliffen: Trotz vier verschiedener Betriebssysteme haben wir eine völlig einheitliche Authentifizierung und Rechteerhöhung. Deswegen möchte ich Ihnen auch mal ein Inventory für die folgende hypothetische, etwas »rauere« Umgebung zeigen:

- ▶ ein Debian-10-System, das direkten Root-Zugang mit SSH-Schlüssel erlaubt
- ▶ ein CentOS-7-System, das den Zugang über den Account `ansible` mit dem Passwort `Supergeheim22!` gestattet. Die Rechteerhöhung geschieht mittels `su`, und wir haben nur Python 2 zur Verfügung.
- ▶ ein Ubuntu-18.04-System, das den Zugang für den User `user1` mit dem SSH-Schlüssel aus der Datei `keys/ubuntu` erlaubt. Der User hat uneingeschränkte `sudo`-Rechte, und sein Passwort lautet `vmware`.

Mit der Inventory-Datei aus Listing 28.11 könnten Sie diese Anforderungen abdecken:

```
[debian_hosts]
debian10  ansible_host=192.168.150.10  ansible_user=root

[centos_hosts]
centos7  ansible_host=192.168.150.20

[ubuntu_hosts]
ubuntu1804  ansible_host=192.168.150.40

[centos_hosts:vars]
ansible_user=ansible
ansible_password=Supergeheim22!
ansible_become=yes
ansible_become_method=su
ansible_become_pass=RootPassw0rd
ansible_python_interpreter=/usr/bin/python2

[ubuntu_hosts:vars]
ansible_user=user1
ansible_ssh_private_key_file=keys/ubuntu
```

```
ansible_become=yes
ansible_become_pass=vmware
```

```
[all:vars]
ansible_python_interpreter=/usr/bin/python3
ansible_ssh_common_args='-o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null'
```

Listing 28.11 »inventory«: Ein komplexeres Inventory

Hauptsächlich sind hier folgende Erkenntnisse zu gewinnen:

- ▶ Der Einsatz von Gruppen zur Parametrisierung ist meist vorteilhaft.
- ▶ Die Parametrisierung von Einzelhosts sollte man sich hingegen weitestgehend ersparen. IP-Adressen sind natürlich eine individuelle Angelegenheit, und für die eine `ansible_user`-Einstellung beim Debian-System hätte sich momentan keine Gruppe gelohnt. Bei zwei oder mehreren solcher Systeme hätte das schon wieder anders ausgesehen.
- ▶ Speziellere Einstellungen übertrumpfen allgemeinere Einstellungen (hier zu sehen bei `ansible_python_interpreter`). Host-spezifische Parameter wirken natürlich am stärksten.

28.2.7 Alternative bzw. mehrere Inventories

Wenn die Menge der mit Ansible verwalteten Systeme anwächst, werden Sie feststellen, dass es oft nicht mehr ganz passend ist, alle Systeme in einer einzigen Inventory-Datei zu verwalten. Ein typisches Beispiel sind verschiedene Bereitstellungsumgebungen wie Development, Testing oder Production, wobei Sie aber z. B. stets alle Maschinen einer Umgebung mit dem Ansible-Bezeichner »all« ansprechen möchten. Darüber hinaus gibt es in Ansible sogar die Möglichkeit *dynamischer Inventories*, die Sie ganz zwangsläufig getrennt von den statischen Inventories behandeln müssen.

In erster Ausbaustufe könnten Sie einfach verschiedene Inventory-Dateien bereitstellen. Natürlich kann es nur maximal ein Default-Inventory geben (das Sie in der `ansible.cfg` festgelegt haben), aber bei jedem geeigneten Ansible-Kommando können Sie mit dem Schalter `-i` ein alternatives Inventory angeben, z. B.:

```
$ ansible all -i production-inv -m ping
```



Anmerkung

Erinnern Sie sich noch an die allerersten Testaufrufe, als Sie noch keinerlei Konfiguration hatten? Etwa:

```
$ ansible all -i debian, -m ping
```


Der Schalter `-i` erlaubt es auch, die Liste der Zielhosts kommasepariert direkt auf der Kommandozeile anzugeben. Wenn man aber nur einen einzigen Host angeben möchte, würde Ansible ohne das Komma am Ende den Hostnamen für einen Dateinamen halten und der Aufruf würde natürlich fehlschlagen.

Eine weitere Möglichkeit zur Auswahl eines Inventorys besteht im Setzen der Umgebungsvariablen `ANSIBLE_INVENTORY`:

```
$ export ANSIBLE_INVENTORY=production-inv
$ ansible [...]
```

In Abschnitt 28.6 werden Sie unter anderem die `host_vars`- bzw. `group_vars`-Parametrisierung kennenlernen; dabei handelt es sich um eine Möglichkeit zur Parametrisierung von Hosts, die über gewisse Verzeichnisse parallel zum jeweiligen Inventory realisiert wird. Spätestens in dem Fall wäre es sinnvoll, verschiedene Inventorys in einer Verzeichnishierarchie zu organisieren – etwa so:

```
ansible/
|-- inventories/
|   |-- devel/
|   |   |-- group_vars/
|   |   |-- host_vars/
|   |   `-- inventory
|   |-- testing/
|   |   |-- group_vars/
|   |   |-- host_vars/
|   |   `-- inventory
|   `-- production/
|       |-- group_vars/
|       |-- host_vars/
|       `-- inventory
|-- [...]
```

Mehrere Inventorys simultan verwenden

Es ist auch möglich, mehrere Inventorys simultan zu verwenden. Dazu wenden Sie die `-i`-Option einfach wiederholt an:

```
$ ansible -i <inventory_1> -i <inventory_2> [...]
```

Wenn Sie mit der Umgebungsvariablen arbeiten, sind die Inventorys kommasepariert anzugeben:

```
$ export ANSIBLE_INVENTORY=<inventory_1>,<inventory_2>[,...]
```

Leerzeichen müssen hier natürlich vermieden werden (oder die rechte Seite muss entsprechend quotiert werden). Es ist sogar möglich, als Inventory ein übergeordnetes Verzeichnis anzugeben, das beliebige Inventories in beliebiger Hierarchietiefe enthält. Ansible bildet dann automatisch die »Summe« über alle enthaltenen Inventories.



Achtung

Achten Sie unbedingt darauf, in einem solchen Gemisch noch den Überblick zu behalten!

28.3 Ad-hoc-Kommandos und Patterns

Das spontane, interaktive Verwenden einer Ansible-Funktionalität über das Kommando `ansible` bezeichnet man als *Ad-hoc-Kommando* oder *Ad-hoc-Aufruf*. In der späteren Praxis wird dies wahrscheinlich zur Nebensache, da Sie dann fast immer mit *Playbooks* und dem entsprechenden Kommando `ansible-playbook` arbeiten werden. Dennoch ist es eine nützliche Möglichkeit, die Sie auf jeden Fall kennen sollten. *Patterns* dienen zum Adressieren von Hosts und sind generell wichtig und nützlich.

28.3.1 Ad-hoc-Kommandos

Sie kennen schon mindestens einen Ad-hoc-Aufruf, nämlich den Ping-Test:

```
$ ansible all -m ping
```

Dabei ist `ping` eines der unzähligen Ansible-Module, und `-m` ist einfach die Kurzform von `--module-name`. Der Bezeichner `all` ist schließlich das Pattern, mit dem die Menge der Hosts festgelegt wird, auf die sich das Ad-hoc-Kommando bezieht. Ein Ad-hoc-Kommando ist also nichts anderes als der Aufruf genau eines Moduls, gegebenenfalls mit einer Parametrisierung. Die allgemeine Form eines Ad-hoc-Kommandos lautet:

```
ansible <PATTERN> -m <MODUL> -a "[<MODUL-PARAMETER>]"
```

Das »command«-Modul

Das `command`-Modul ist an dieser Stelle sicher eines der nützlichsten; sein Parameter ist ein auszuführendes Linux-Kommando. Lassen Sie sich beispielsweise einmal die Belegung der Root-Partition auf allen Hosts anzeigen:

```
$ ansible all -m command -a "df -h /"
debian | CHANGED | rc=0 >>
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/debian--11--vg-root 62G  1.1G   58G   2% /
[...]
```

Das `command`-Modul ist übrigens sogar der Default; der folgende Aufruf würde also dasselbe Ergebnis bringen:

```
$ ansible all -a "df -h /"
```

Wenn Sie pro Host ein einzeliges Ergebnis erwarten, kann die Option `-o` / `--one-line` die Weiterverarbeitung erleichtern oder auch nur die Übersicht erhöhen:

```
$ ansible all -a "hostname" -o
debian | CHANGED | rc=0 | (stdout) debian
[...]
```

Das »shell«-Modul

Wenn Sie in Ihrem Aufruf Shell-Mechanismen wie `»|«`, `»>«` usw. benötigen, müssen Sie zum `shell`-Modul greifen, da `command` das übergebene Kommando ohne Umweg über eine Shell direkt zur Ausführung bringt:

```
$ ansible all -m shell -o -a "ps -ef | wc -l"
debian | CHANGED | rc=0 | (stdout) 91
[...]
```

28.3.2 Use Cases jenseits von »command« und »shell«

Sehen Sie hier nun einige Beispiele für Anwendungen von Ad-hoc-Kommandos in der Praxis.

- ▶ Hosts neu starten:

```
$ ansible all -m reboot
```

Dieses Modul ist erst in Ansible 2.7 hinzugekommen. Falls Sie noch eine ältere Version einsetzen, müssen Sie sich mit einem simplen `ansible all -a /sbin/reboot` behelfen. Das Verhalten ist damit allerdings ein anderes, da Ihnen sofort einfach alle Verbindungen wegbrechen und Sie nicht erfahren, wann die Hosts wieder online sind.

- ▶ Ein Verzeichnis anlegen:

```
$ ansible all -m file -a "dest=/opt/test1 state=directory"
```

- ▶ Einen User anlegen:

```
$ ansible all -m user -a "name=john shell=/bin/bash"
```

- ▶ Ein Paket installieren:

```
$ ansible all -m package -a "name=tree"
```

Natürlich vorausgesetzt, das Paket hat auf allen Systemen denselben Namen.

- ▶ Eine Datei hochladen:

```
$ ansible all -m copy -a "src=/etc/passwd dest=/tmp/test.txt"
```

Woher soll man das alles wissen?

(Falls Sie sich das gerade gefragt haben.) Es sind dieselben Module, die Sie bald tagtäglich verwenden werden, und die wichtigsten Parameter kennt man irgendwann auswendig. In Abschnitt 28.7 geben wir auch eine Kurzübersicht über die gängigsten Module, aber in der Praxis werden Sie wahrscheinlich meist mit der offiziellen Dokumentation arbeiten. Der schnellste Weg dahin ist (wie so oft) Google bzw. die Suchmaschine Ihrer Wahl; versuchen Sie beispielsweise einmal »ansible user module« oder »ansible copy module«. Der erste Treffer ist meistens auch schon der richtige – jedenfalls sofern er auf irgendetwas unterhalb von <http://docs.ansible.org/> verweist.

28.3.3 Idempotenz

Eine Eigenschaft, auf die bei Ansible sehr großer Wert gelegt wird, ist die *Idempotenz*. Dieser Begriff kommt ursprünglich aus der Mathematik und bezeichnet dort eine Eigenschaft von Funktionen:

$$f * f = f$$

Etwas salopp gesagt bedeutet dies, dass es egal ist, ob man eine Funktion einmal oder zweimal ausführt – wiederholtes Ausführen ändert nichts (mehr) am Ergebnis. Beispiele in der Mathematik sind etwa Projektionen oder Betragsfunktionen.

Für ein Konfigurationsmanagementsystem bedeutet dies: *Wenn ein Konfigurationszustand bereits erreicht ist, dann sind alle weiteren Ausführungen der entsprechenden Anweisung ohne Effekt.* (Insbesondere gibt es keine negativen Effekte wie Fehlermeldungen.)

Beispiel: Ein Softwarepaket soll installiert werden. Der erste Aufruf erledigt das; weitere identische Aufrufe stellen nur noch fest, dass nichts mehr zu tun ist. Das klingt banal, trifft aber auf nahezu alle Ansible-Funktionen zu:

- ▶ Eine Konfigurationsdatei enthält schon eine gewünschte Zeile – es ist nichts mehr zu tun.
- ▶ Ein symbolischer Link ist schon wie gewünscht vorhanden – es ist nichts mehr zu tun.
- ▶ Ein Benutzer ist schon in einer Gruppe – es ist nichts mehr zu tun.

Dieses sehr angenehme Verhalten würde man mit einer selbst geskripteten Lösung niemals in dieser Konsequenz erreichen. Idempotenz ist also eine der vielen Annehmlichkeiten von Ansible!

28.3.4 Interne Funktionsweise

Zum Verständnis mancher Begriffe ist es hilfreich, ein wenig hinter die Kulissen von Ansible zu schauen. Jedoch ist das momentan kein Pflichtwissen – Sie könnten Ansible hier weitestgehend auch als Blackbox betrachten und dennoch genauso erfolgreich damit arbeiten.

Parallele Ausführung

Ansible greift pro Task grundsätzlich parallel auf seine Hosts zu. Die Anzahl der *Forks* (also der gleichzeitigen Verbindungen) ist mit dem Default-Wert von 5 allerdings eher konservativ bemessen.

Sie können den gewünschten Wert entweder dynamisch mit der Option `-f` auf der Kommandozeile angeben:

```
$ ansible all -f 10 -m ping
```

Oder ihn dauerhaft in der Konfiguration im Abschnitt `[defaults]` setzen:

```
# [defaults]
forks = 10
```

Listing 28.12 »ansible.cfg«: Ausschnitt

In der Praxis sieht man durchaus Werte wie 50 oder mehr. Hauptsache, Ihr Control Host und Ihr Netzwerk halten das aus.

Persistente Verbindungen

Sie wissen bereits, dass Ansible seine Hosts standardmäßig über SSH anspricht. Um hier eine adäquate Performance zu erreichen (der Auf- und Abbau von SSH-Verbindungen ist durchaus ein Faktor!), arbeitet Ansible mit *persistenten Verbindungen*.

Lassen Sie uns diese einmal sichtbar machen. Das geht beispielsweise mit Tools wie `netstat`, `ss` oder `lsof`; eines davon wird Ihnen sicherlich zur Verfügung stehen. Wichtig ist es, zunächst eine Ansible-Aktion auf den Zielhosts durchzuführen, da die Verbindungen nach 30 Sekunden Inaktivität wieder weggeräumt werden:

```
$ ansible all -m ping
[...]
```

```
$ netstat -tan
```

```
[...]
tcp 0 0 192.168.150.100:55968 192.168.150.10:22 ESTABLISHED
tcp 0 0 192.168.150.100:48398 192.168.150.20:22 ESTABLISHED
tcp 0 0 192.168.150.100:58532 192.168.150.30:22 ESTABLISHED
tcp 0 0 192.168.150.100:48866 192.168.150.40:22 ESTABLISHED
[...]
```

```
$ lsof -i :22
```

```
[...vergleichbare Ausgabe...]
```

```
$ ss -tn
```

```
[...vergleichbare Ausgabe...]
```

```
$ ls -l ~/.ansible/cp/
srw----- 1 vagrant vagrant 0 Oct 31 12:00 4da0326269
srw----- 1 vagrant vagrant 0 Oct 31 12:00 8082b77a5c
srw----- 1 vagrant vagrant 0 Oct 31 12:00 8c04ad47d2
srw----- 1 vagrant vagrant 0 Oct 31 12:00 c465e60c60
```

(Die Ausgaben der Kommandos wurden aus Platzgründen teilweise etwas modifiziert.)

Das abschließende Verzeichnislisting zeigt Ihnen die von SSH verwendeten Socket-Dateien («cp» steht für »ControlPath«).

Hängende Verbindungen

In ganz seltenen Fällen können die persistenten Verbindungen auch schon mal Probleme bereiten. Ein häufiger Grund ist, dass man sich mit Ansible an den Firewall-Einstellungen des Zielsystems zu schaffen gemacht hat und die Verbindungen nun quasi tot sind. Das äußert sich dann so, dass Ansible vom entsprechenden Zielhost einfach keine Antwort mehr bekommt, obwohl netzwerktechnisch (normales Ping, SSH) alles in Ordnung ist.

Abhilfe schafft in dem Moment entweder ein kurzes Warten, oder Sie machen die hängenden SSH-Prozesse ausfindig und beenden sie:

```
$ lsof -i :22
COMMAND  PID  USER  FD  TYPE  DEVICE  SIZE/OFF  NODE NAME
[...]
ssh      12314  user1  3u  IPv4  1249038      0t0  TCP  [...]-><host>:ssh
[...]
```

```
$ kill -HUP 12314
```

Aber da ungenutzte Verbindungen standardmäßig nach 30 Sekunden sowieso weggeräumt werden, wäre einfaches Warten wahrscheinlich effizienter gewesen.

28.3.5 Die Ansible-Konsole

Wenn Sie für längere Zeit interaktiv mit diversen Target Hosts arbeiten wollen, können Sie das Kommando `ansible-console` nutzen, das quasi eine integrierte Shell-Umgebung zur Verfügung stellt:

```
$ ansible-console
oder falls es Farbprobleme gibt:
$ ANSIBLE_NOCOLOR=1 ansible-console
```

Das erste Wort auf jeder Kommandozeile wird als Ansible-Modul interpretiert. Existiert kein solches Modul, so wird die ganze Zeile stattdessen mit dem Shell-Modul als Linux-Kommando ausgeführt:

```
vagrant@all (4)[f:5]$ ping
[...]
```

```
vagrant@all (4)[f:5]$ df -h /
[...]
```

- ▶ Sie verlassen die Konsole mit `exit` oder dem UNIX-üblichen `Strg`+`D`.
- ▶ In den seltenen Fällen, in denen Sie ein Kommando ausführen wollen, das genauso heißt wie ein Modul (z.B. `hostname`), schreiben Sie am einfachsten `command` oder `shell` davor!
- ▶ Mit dem eingebauten `cd`-Kommando können Sie in einzelne Target Hosts oder -Gruppen (oder in generell jedes Pattern) »wechseln«. Ein `cd` ohne Argument weitet die Adressierung wieder auf alle Hosts aus.
- ▶ Mit dem eingebauten `list`-Kommando können Sie sich anzeigen lassen, welche Hosts Sie momentan adressieren.

Farbgebung

Wenn Sie gerne mit weißen Terminalfenstern arbeiten, so ist die standardmäßige weiße Schrift der Konsole etwas ungünstig. Seit Version 2.7 können Sie in der Konfigurationsdatei im Abschnitt `[colors]` die Schriftfarbe der Eingabeaufforderung auch festlegen. Ergänzen Sie also bei Bedarf etwas wie:

```
[colors]
console_prompt = black
```

Listing 28.13 »ansible.cfg«: Ausschnitt zur Anpassung der Schriftfarbe der Ansible-Konsole

28.3.6 Patterns zum Adressieren von Hosts

Bislang haben wir in den meisten Fällen mit dem Bezeichner `all` alle Hosts aus unserem Inventory angesprochen. Mit Ansible können Sie Ihre Aktionen natürlich auch an einzelne Hosts oder Gruppen richten:

```
$ ansible suse -m ping
$ ansible test_hosts -m ping
```

Wie bereits erwähnt wurde, ist ja `all` auch nur eine Gruppe, die stets per Default zur Verfügung steht, ohne dass man sie eigens definieren müsste.

Mittels gewisser Sonderzeichen stehen auch andere Auswahlmöglichkeiten zur Verfügung. Beispielsweise steht der übliche Joker `»*«` zur Verfügung, den Sie wegen der Shell aber quotieren sollten:

```
$ ansible '*' -m ping
$ ansible 'su*' -m ping
```

Mit Komma oder Doppelpunkt können Sie Vereinigungsmengen von Hosts oder Gruppen bilden:

```
$ ansible 'debian,rocky' -m ping
```

Folgt vor einem Pattern ein Ausrufezeichen, so ist das als Ausschluss zu verstehen. Ausschlüsse können auch mehrfach angewendet werden – beispielsweise wie folgt:

Alle außer suse:

```
$ ansible 'all:!suse' -m ping
```

Alle außer suse und ubuntu:

```
$ ansible 'all:!suse:!ubuntu' -m ping
```

Für ganz ausgefuchste Anwender stehen auch reguläre Ausdrücke zur Verfügung. Das Pattern muss dann mit einer Tilde (~) beginnen:

```
$ ansible '~^ub.*u$' -m ping
```

Dieses exemplarische Pattern passt auf alle Hosts, deren Name mit »ub« beginnt und mit »u« endet. Sämtliche Möglichkeiten sind unter http://docs.ansible.com/ansible/latest/user_guide/intro_patterns.html beschrieben.

28.4 Die Konfigurations- und Serialisierungssprache YAML

In diesem Abschnitt geht es um die Sprache YAML – aus pragmatischer Sicht. Diese Sprache müssen Sie zunächst lernen, wenn Sie produktiv mit Ansible (oder Docker Compose oder Kubernetes oder ...) arbeiten wollen. YAML ist eine syntaktisch recht einfache Sprache zur Datenserialisierung, die meist als »Konfigurationssprache« genutzt wird. Die Abkürzung YAML bedeutet *YAML Ain't Markup Language* (ursprünglich: *Yet Another Markup Language*).

28.4.1 Syntax und Struktur

YAML weist hinsichtlich Syntax und Struktur die folgenden Merkmale auf:

- ▶ YAML-Dateien tragen per Konvention die Dateierweiterung `.yaml` oder `.yml`.
- ▶ YAML-Dateien sind in UTF-8 oder UTF-16 codiert.
- ▶ Kommentare sind mit dem üblichen `#`-Zeichen möglich.
- ▶ Die inhaltliche Struktur eines YAML-Dokuments wird in erster Linie durch *Einrückung* definiert.
- ▶ Es können Einzelwerte (Skalare), Listen (Arrays) und Maps (assoziative Arrays/Dictionaries) und beliebig verschachtelte Kombinationen davon abgebildet werden.
- ▶ Skalare Textwerte können (falls möglich) einfach so notiert werden; bei Bedarf ist auch die Quotierung mit `"..."` oder `'...'` möglich.

Ein erstes Beispiel dafür, wie YAML sich »anfühlt«, mag die Webseite des YAML-Projekts liefern (siehe <http://yaml.org>). Sie präsentiert sich in syntaktisch korrektem YAML.

Die drei Minuszeichen (`---`), die man zu Beginn einer YAML-Datei oft sieht, signalisieren einerseits den Beginn eines neuen Dokuments und erlauben es andererseits, darüber gewisse Metainformationen, z. B. die verwendete YAML-Version, zu spezifizieren. Letzteres ist in der Praxis fast immer unnötig, und auch die drei Minuszeichen werden sehr oft weggelassen. Wenn Ihre Datei allerdings mehrere Dokumente enthalten soll, dann werden sie als Trenner verpflichtend. In Ansible z. B. wird diese Möglichkeit überhaupt nicht genutzt, in Kubernetes hingegen sehr wohl. Es gibt auch drei Punkte (`...`), um das Ende des Dokuments zu markieren, aber auch darüber können wir getrost erst mal hinwegsehen.

28.4.2 YAML-Files editieren

Für anspruchsvollere YAML-Dokumente ist ein Editor mit YAML-Support empfehlenswert, der nicht nur Syntax-Highlighting, sondern auch eine möglichst intelligente Einrückhilfe bieten sollte. Einige Kandidaten möchten wir Ihnen kurz vorstellen.

vim

Für viele Admins ist der Editor `vim` das Lieblingswerkzeug, aber so richtig YAML-tauglich ist er im Auslieferungszustand meist noch nicht. Für das allgemeine und produktive Arbeiten mit YAML-Dateien sind daher einige Anpassungen zu empfehlen:

- ▶ Das Syntax-Highlighting ist manchmal fehlerhaft bzw. nicht intelligent genug umgesetzt. Falls Sie das stört und Sie überhaupt Farben sehen möchten, kommentieren Sie in der Datei `/usr/share/vim/vim*/syntax/yaml.vim` die zwei Bereiche aus, die mit `syn region yamlFlowString` beginnen (es sind sechs Textzeilen in Folge). Damit sehen Sie zwar weniger Farben, aber dafür werden sie an den richtigen Stellen eingesetzt. Achtung: Das Kommentarzeichen ist hier das doppelte Hochkomma!
- ▶ Hier einige sinnvolle Einstellungen für Ihre persönliche `~/vimrc`:

```
filetype plugin indent on
syntax on
set pastetoggle=<F2>
autocmd FileType yaml setlocal tabstop=2 softtabstop=2 shiftwidth=2
\ expandtab autoindent
```

Listing 28.14 »~/vimrc«: vim-Konfiguration für YAML-Dateien ohne spezielles Plug-in

Im Wesentlichen geht es hier darum, möglichst konsistent eine Einrücktiefe von zwei Leerzeichen zu konfigurieren, wobei die Einzugschritte problemlos mit der Tabulator- bzw. Backspace-Taste gewechselt werden können. Falls Sie sich während des Tippens an zu viel spontaner Einrück-Intelligenz stören, dann verzichten Sie probeweise in der ersten Zeile auf das Wort `indent`.

Es gibt für `vim` auch spezielle Plug-ins, die über den allgemeinen YAML-Support hinaus auch die jeweilige »Sprache« unterstützen. Bei Ansible-Anwendern recht bekannt ist z. B. das Plug-in `ansible-vim` (<https://github.com/pearofducks/ansible-vim>). Absolute `vim`-Experten, die so etwas ausprobieren möchten, werden sicher mit den Erklärungen auf der Projektseite zu recht kommen. Zur Arbeit mit `vim` abschließend noch ein Tipp: Falls Ihnen doch einmal irgendwelche unsichtbaren Tabulator- oder Leerzeichen Ärger machen, dann ist ein Umschalten auf `:set list` sehr hilfreich, um diese nicht druckbaren Zeichen sichtbar zu machen.

nano

Der beliebte `nano`-Editor bietet leider meist keinen *out of the box*-Support für YAML. Aktuell im Umlauf befindliche Versionen bringen meist nicht einmal ein Syntax-Highlighting für YAML mit. Auf GitHub finden sich aber durchaus Syntax-Highlighting-Dateien, die Sie dann mittels `include` in eine `nanorc`-Datei einbinden könnten. Auf einem Debian-System ist das Einbinden aller Highlighting-Dateien unter `/usr/share/nano/` schon in der globalen `nano`-Konfiguration vorbereitet. Um etwas Farbe ins Spiel zu bringen, würde also beispielsweise dieser Befehl ausreichen:

```
$ sudo wget https://raw.githubusercontent.com/serialhex/nano-highlight/\
master/yaml.nanorc -O /usr/share/nano/yaml.nanorc
```

Jetzt fehlt nur etwas Unterstützung beim korrekten Einrücken von YAML-Code. Hier einige sinnvolle Einstellungen, die Sie in Ihre persönliche `~/nanorc` übernehmen können:

```
set autoindent
set tabsize 2
set tabstospaces
```

Listing 28.15 »`~/nanorc`«: exemplarische `nano`-Konfiguration für YAML-Dateien

Mit `autoindent` bleibt man auf derselben Einzugsebene, wenn man eine neue Zeile beginnt; die anderen Direktiven sind selbsterklärend. Als einziger Haken bleibt, dass es keine Möglichkeit gibt, diese Einstellungen *nur mit YAML-Dateien* zu verknüpfen. Sollte Sie das stören, könnten Sie alternativ zur dieser `~/nanorc` den `nano` im Falle von YAML-Dateien auch immer mit `nano -iET2 <DATEI.yml>` aufrufen oder sich daraus ein Alias wie `yamlnano` basteln ...

28.4.3 Listen und Maps

Wir geben nun kurze Beispiele für die wesentlichen syntaktischen Elemente von YAML.

Listen

Listen – also geordnete Mengen von Dingen – werden in YAML wie folgt dargestellt:

- Hund
- Katze

```
- Maus
```

```
# Dasselbe mit Inline-Notation:
```

```
[Hund, Katze, Maus]
```

Listing 28.16 Darstellung von Listen in YAML

Maps (assoziative Arrays)

Maps (assoziative Arrays) sind eine Sammlung von Schlüssel-Wert-Paaren. Sie werden wie folgt dargestellt:

```
Hund: Wuff
```

```
Katze: Miau
```

```
Maus: Fiep
```

```
# Dasselbe mit Inline-Notation:
```

```
{Hund: Wuff, Katze: Miau, Maus: Fiep}
```

Listing 28.17 Darstellung einer Map in YAML

28.4.4 Verschachtelte Strukturen

In YAML sind beliebige Kombinationen von Maps und Listen möglich. **Beachten Sie dabei unbedingt die Einrückung!** Die genaue Einrücktiefe ist dabei unwichtig. Üblich sind zwei Leerzeichen pro Ebene.

► Liste von Listen:

```
-
  - Waldi
  - Bello
-
  - Mimi
  - Kitty
```

Listing 28.18 Eine Liste von Listen in YAML

Eine solche Kombination werden Sie aber relativ selten zu Gesicht bekommen. In der Praxis gibt man Listen von Maps immer den Vorzug.

► Liste von Maps:

```
- Hund: Bello
  Laut: Wuff

- Katze: Kitty
  Laut: Schnurr
```

Listing 28.19 Eine Liste von Maps in YAML

► Map mit Listen:

```
Bello:  
  - Wuff  
  - Knurr
```

```
Fiffi:  
  - Kläff  
  - Jaul
```

Listing 28.20 Eine Map mit Listen in YAML

► Map mit Maps:

```
Hund:  
  Name: Bello  
  Alter: 7  
  Futter: Pedigree
```

```
Katze:  
  Name: Mimi  
  Alter: 9  
  Futter: Whiskas
```

Listing 28.21 Eine Map mit Maps in YAML

28.4.5 Textpassagen und Block-Ausdrücke

Möchten Sie Texte mit Zeilenumbrüchen in YAML spezifizieren, so steht Ihnen mit den *Block-Ausdrücken* eine interessante Möglichkeit zur Verfügung.

Hier ein typisches Beispiel – eine längere Textpassage als Map-Wert:

```
brief: |  
  Sehr geehrter Herr XY,  
  
  ich hoffe, es geht Ihnen gut!  
  
  Mit freundlichen Grüßen  
  Axel Miesen
```

Listing 28.22 Ein Block-Ausdruck in YAML

An der ersten Zeile orientiert sich der ganze Folgetext, Einrückungen bleiben erhalten.

Die Variante mit »»« ersetzt alle Newlines zwischen direkt aufeinanderfolgenden Zeilen (außer dem letzten) durch ein Leerzeichen:

```
text: >
  Dies ist einfach nur ein sehr langer Satz,
  der aber mittendrin nicht umbrochen werden soll!
```

Listing 28.23 Ein Block-Ausdruck ohne Umbruch in YAML

Es existieren auch Variationen dieser Block-Ausdrücke, mit denen Sie das abschließende Newline und darauf folgende Leerzeilen kontrollieren können (siehe Tabelle 28.2).

Ausdruck	Bedeutung
oder >	Abschließendes Newline erhalten, abschließende Leerzeilen entfernen.
- oder >-	Beides entfernen.
+ oder >+	Beides erhalten.

Tabelle 28.2 Varianten von YAML-Block-Ausdrücken

28.4.6 Das Nichts in YAML

Ebenso wie in vielen anderen Sprachen gibt es auch in YAML die Möglichkeit, einen nicht vorhandenen Wert zu spezifizieren. Möglicherweise kennen Sie bereits den NULL-Wert in SQL, den undef-Wert in Perl oder den None-Wert in Python. Falls Sie mit dem Konzept noch nicht vertraut sind, stellen Sie sich einfach vor, Sie messen zu bestimmten Zeitpunkten Temperaturen und zwischendurch fällt immer wieder mal der Sensor aus, sodass nichts gemessen wird. In der Liste der Messwerte möchten Sie dann zu diesen Zeitpunkten aber auch festhalten, dass *kein* Wert gemessen wurde. Rein syntaktisch haben Sie in YAML sogar mehrere Möglichkeiten, dies auszudrücken:

```
wert1: # Dieser Kommentar dient nur der Lesbarkeit
wert2: ~
wert3: null
wert4: Null
wert5: NULL
```

Listing 28.24 Nicht vorhandene Werte in YAML

28.5 Playbooks und Tasks: die Grundlagen

Playbooks sind der Einstieg in jedes Ansible-Projekt. Im einfachsten Fall enthalten sie eine geordnete Menge von einzelnen Schritten (*Tasks*), die beschreiben, wie die gewünschten Konfigurationsziele für die Target Hosts zu erreichen sind.

28.5.1 Hallo Ansible – das allererste Playbook

Obwohl Ansible keine Programmiersprache im engeren Sinne ist, wollen wir doch der guten alten Tradition folgen und als erste Amtshandlung einmal ein »Hallo Welt«-Playbook erstellen. Playbooks werden in YAML verfasst und tragen per Konvention die Dateierweiterung `.yaml` oder `.yml`. Das Playbook aus Listing 28.25 ist so ziemlich das einfachste, das überhaupt möglich ist:

```
---
- hosts: localhost

  tasks:
    - debug: msg="Hallo Ansible!"
```

Listing 28.25 »hallo-ansible.yml«: ein allererstes Playbook



Achtung

Falls Sie das Kapitel bis hierher nicht linear durchgearbeitet haben, sollten Sie entweder das YAML-Format hinreichend gut kennen oder *wirklich* erst Abschnitt 28.4 lesen.

Das Wesentliche bei YAML ist die korrekte Einrückung der Textbausteine – diese darf auf keinen Fall einfach irgendwie geschehen!

Damit unser Projektordner über kurz oder lang nicht zu unübersichtlich wird, schlage ich vor, für Playbooks einen Ordner `playbooks/` anzulegen:

```
$ cd ~/ansible/projects/start; mkdir playbooks
```

Speichern Sie die Playbook-Datei in diesen neuen Ordner. Bevor wir das Ganze starten, wollen wir zunächst den Inhalt klären:

- ▶ Die drei Minuszeichen sind bekanntlich eine YAML-Markierung, die den Beginn des Dokuments kennzeichnen. Viele Ansible-Anwender lassen sie weg, was kein Problem ist, da mehrere YAML-Dokumente in einer Datei unter Ansible sowieso nicht möglich sind. Nur der Vollständigkeit halber sei erwähnt: Ansible nutzt als Parser das Python-PyYAML-Modul, das zurzeit die YAML-Version 1.1 implementiert.
- ▶ Das Schlüsselwort `hosts` gibt an, welche Target Hosts Sie ansprechen wollen. Anstelle von `localhost` wäre auch ein beliebiges Pattern (vergleiche Abschnitt 28.3.6) möglich. Sehr oft sieht man hier auch den Gruppennamen `all`, um alle Inventory-Hosts anzusprechen.

Auch eine Liste von Patterns wäre möglich, z. B.:

```
- hosts:
  - debian
  - rocky
```

Oder in der kompakten YAML-Listenform:

```
- hosts: [debian, rocky]
```

Oder eben sogar als Ansible-Pattern:

```
- hosts: debian, rocky
```

- ▶ Das Schlüsselwort `tasks` zeigt an, dass nun die Arbeitsschritte folgen, die die Target Hosts in den gewünschten Zielzustand überführen sollen.
- ▶ Der Aufruf des `debug`-Moduls ist momentan unser einziger Task; mit dem Parameter `msg` kann man die gewünschte Ausgabe bestimmen.

Auf Anhieb ist wahrscheinlich nicht völlig klar, warum das Playbook mit einer Liste beginnt. (YAML-technisch gesehen, ist der Inhalt unseres Playbooks eine Liste mit einem einzigen Element!) Die Erklärung ist folgende: Ein Playbook kann sich aus mehreren sogenannten *Plays* zusammensetzen – diese bestehen jeweils aus einer Menge von Hosts, denen gewisse Tasks zugeordnet sind. Playbooks mit mehr als einem Play werden Sie aber erst sehr viel später benötigen. Um das Ganze (endlich) zu starten, verwenden Sie das Kommando `ansible-playbook` und übergeben den Pfad zum Playbook als Parameter. Wenn Ihr aktuelles Arbeitsverzeichnis also der Wurzelordner unseres Projekts ist, dann sieht das so aus:

```
$ ansible-playbook playbooks/hallo-ansible.yml
```

```
PLAY [localhost] *****

TASK [Gathering Facts] *****
ok: [localhost]

TASK [debug] *****
ok: [localhost] => {
  "msg": "Hallo Ansible!"
}

PLAY RECAP *****
localhost                : ok=2    changed=0    unreachable=0    failed=0
                          skipped=0    rescued=0    ignored=0
```

Auch die Ausgabe hat etwas Klärungsbedarf:

- ▶ Die Ausgabe wird zunächst mit `PLAY` und der Auflistung der entsprechenden Zielsysteme (in unserem Beispiel lediglich `localhost`) eingeleitet. Sie können ein Play auch mit einem `name`-Attribut versehen, dann würde an dieser Stelle der Wert dieses Attributs ausgegeben. Wenn das Playbook nur ein einziges Play enthält, macht man das in der Regel aber nicht.

- ▶ Es folgen nun alle Tasks mit einer kurzen Beschreibung und einem entsprechenden Ergebnis.
- ▶ Den Task `Gathering Facts` haben wir gar nicht bestellt, aber es ist das Default-Verhalten von Ansible, zu Beginn eines Plays erst einmal Informationen über die beteiligten Zielsysteme einzusammeln. Wir kommen darauf in Abschnitt 28.6.3 noch ausführlich zu sprechen. In den meisten realen Playbooks braucht man diese Fakten auch; in unserem einfachen Beispiel könnte man durch die Angabe von `gather_facts: no` oder `gather_facts: false` auch auf das Einsammeln verzichten und so etwas Zeit sparen:


```
- hosts: localhost
  gather_facts: no
```
- ▶ Die letzte Zeile liefert schließlich einen Abschlussbericht: Auf dem Zielsystem `localhost` wurden zwei Tasks mit dem gewünschten (erfolgreichen) Ergebnis beendet (`ok=2`), es gab keine Änderungen (`changed=0`), das Zielsystem war erreichbar (`unreachable=0`), und es schlugen keine Tasks fehl (`failed=0`). Die weiteren Felder `skipped`, `rescued` und `ignored` können Sie zunächst getrost ignorieren.

Nun ist `localhost` sicher ein nicht so häufiges Ziel für Verwaltungsaufgaben. Lassen Sie uns stattdessen einmal alle wirklichen Ziele im Testlabor ansprechen:

```
---
- hosts: all

  tasks:
    - debug: msg="Hallo Ansible!"
```

Listing 28.26 »hallo-alle.yml«: Hallo an alle

Und um das Ganze etwas spannender zu machen, fahren Sie doch eine Maschine (z.B. `suse`) vorher mal herunter. Das Ergebnis sieht dann in etwa so aus:

```
$ ansible-playbook hallo-alle.yml
```

```
PLAY [all] *****

TASK [Gathering Facts] *****
ok: [ubuntu]
ok: [debian]
ok: [rocky]
fatal: [suse]: UNREACHABLE! => {"changed": false, "msg":
"Failed to connect to the host via ssh: ssh: connect to host suse port 22:
No route to host", "unreachable": true}

TASK [debug] *****
ok: [debian] => {
```



```

    "msg": "Hallo Ansible!"
  }
ok: [rocky] => {
    "msg": "Hallo Ansible!"
  }
ok: [ubuntu] => {
    "msg": "Hallo Ansible!"
  }

```

```

PLAY RECAP *****
debian                : ok=2    changed=0    unreachable=0    failed=0
rocky                 : ok=2    changed=0    unreachable=0    failed=0
suse                  : ok=0    changed=0    unreachable=1    failed=0
ubuntu                : ok=2    changed=0    unreachable=0    failed=0

```

Hier fällt zunächst auf, dass es pro Task keine fixe Reihenfolge der Zielsysteme zu geben scheint. Durch die parallele Abarbeitung ist das auch tatsächlich ziemlich zufällig; wir hatten dies in Abschnitt 28.3.4 bereits besprochen. Außerdem ist zu bemerken, dass suse komplett aus dem Spiel ausgeschieden ist, nachdem seine Unerreichbarkeit festgestellt wurde, denn der zweite Task hat erst gar nicht mehr versucht, diesen Host anzusprechen.

28.5.2 Formulierung von Tasks

Die allermeisten Tasks bestehen aus dem Aufruf eines Ansible-Moduls, gegebenenfalls mit geeigneter Parametrisierung.

Wichtig

Der Deutlichkeit halber möchten wir dies noch einmal klar festhalten: Sie können pro Task nur *ein* Modul aufrufen! Wenn Ihre Aufgabe mit zwei (oder mehr) Modulaufrufen erledigt werden muss, benötigen Sie zwei (oder mehr) Tasks.

Davon abgesehen, gestattet Ansible bei der Formulierung von Tasks aber durchaus gewisse Freiheiten – umso mehr sollten Sie sich im Team möglichst früh auf einen Style Guide bzw. einige Best Practices einigen. Das Wichtigste ist, dass jeder Task ein `name`-Attribut bekommt. Bei einem `debug`-Task muss man das sicher nicht ganz so eng sehen, aber für die meisten anderen Arten von Tasks sollten Sie es sich zur Regel machen. Eine Leerzeile zwischen den einzelnen Tasks schadet sicher auch nicht. Sehen Sie hier ein Beispiel, wie man es besser nicht macht:

```

tasks: # so eher nicht:
  - debug: msg="Hallo"

```



- command: df -h /
- service: name=sshd state=started

Viel besser wäre es so:

- ```
tasks:
 - name: Eine Test-Ausgabe
 debug: msg="Hallo"

 - name: Platzbedarf der Root-Partition ermitteln
 command: df -h /

 - name: Den SSH-Dienst starten
 service: name=sshd state=started
```

Die Task-Namen erscheinen beim Playbook-Lauf auch in der Ausgabe, was Übersicht und Orientierung immens erleichtert. Der letzte der drei Tasks ist ein typisches Beispiel für einen parametrisierten Modulaufruf, bei dem die Parameter in der Form `<name1=wert1> <name2=wert2> ...` spezifiziert werden können.

Um lange (oder gar überlange) Zeilen zu vermeiden, kommt hier manchmal ein YAML-Block-Ausdruck zum Einsatz:

- ```
- name: Den SSH-Dienst starten  
  service: >  
    name=sshd  
    state=started
```

Alternativ können und sollten Sie die Parameter eher als YAML-Map spezifizieren. Achten Sie dabei penibel auf die Einrückebenen:

- ```
- name: Den SSH-Dienst starten
 service:
 name: sshd
 state: started
```



#### **Tip**

Die Formulierung von Modulparametern als YAML-Map sollte die Methode der Wahl sein – zumindest sobald Sie zwei oder mehr Parameter übergeben möchten!

### **28.5.3 Beenden von Plays**

Wollen Sie (meist zu Testzwecken) ein Play vorzeitig beenden, so steht Ihnen die Anweisung

- meta: end\_play

zur Verfügung, die Sie auf derselben Einrückebene wie einen normalen Task platzieren. Das Play wird dann an dieser Stelle beendet.

#### 28.5.4 Fehlerbehandlung, Retry-Files

Fehler passieren – leider sogar mitunter dann, wenn man selbst alles richtig gemacht hat. In diesem Abschnitt wollen wir klären, wie Ansible mit Fehlern umgeht. Man unterscheidet zwei Arten von Fehlern:

1. Parsezeit-Fehler
2. Laufzeit-Fehler

*Parsezeit-Fehler* sind Fehler, die Ansible entdecken kann, bevor das Playbook überhaupt richtig »losläuft«: Vertipper bei Modul- oder Attributnamen, falsche YAML-Einrückung (sehr beliebt) oder Ähnliches. Dieser Fehlertyp ist eigentlich recht langweilig: Ansible gibt mehr oder weniger hilfreiche Erklärungen aus, Sie beheben den Fehler und versuchen es noch einmal.

Die eigentlich relevanten Fehler sind hingegen die *Laufzeit-Fehler*: Auf irgendeinem Host kann ein Task nicht erfolgreich durchgeführt werden – sozusagen ein schwerer Ausnahmefehler. Standardmäßig verhält sich Ansible bei einem Laufzeit-Fehler wie folgt:

##### Verhalten bei Laufzeit-Fehlern

Wenn ein Task auf einem Host in einen Fehler läuft, wird der entsprechende Host von allen weiteren Aktionen ausgeschlossen. Für die übrigen (fehlerfreien) Hosts geht das Play weiter.



#### Retry-Files

Im Fehlerfall kann Ansible eine Datei namens `<PLAYBOOKNAME>.retry` anlegen, die die Namen aller Failed Hosts enthält. Diese können Sie bei Bedarf bei einem erneuten Playbook-Lauf zum Limitieren verwenden (Option `--limit @<RETRYFILE>`), womit die Menge der Target Hosts gegebenenfalls sehr verkleinert wird. Die Hosts, auf denen bereits alles erfolgreich gelaufen ist, muss man ja schließlich nicht noch einmal behelligen – auch das Feststellen der Idempotenz belastet die Systeme ein wenig. Es gibt zwei Konfigurationsdirektiven, die dieses Verhalten steuern:

- ▶ `retry_files_enabled` legt fest, ob überhaupt Retry-Files erzeugt werden (Default ab Ansible 2.8: `false`, vorher `true`).
- ▶ `retry_files_save_path` legt ein Verzeichnis fest, in das die Retry-Files geschrieben werden (per Default neben dem Playbook).

Unsere Empfehlung: Retry-Files sind grundsätzlich nicht schlecht, nur der Default-Ablageort stört manchmal. Um es zu aktivieren konfigurieren Sie Ansible wie folgt:

```
[defaults]
retry_files_enabled = yes
retry_files_save_path = ~/.ansible/retry-files
```

**Listing 28.27** »ansible.cfg«: Aktivieren von Retry-Files

### 28.5.5 Tags

Mitunter ist es wünschenswert, anstelle eines kompletten Playbooks nur Teile davon laufen zu lassen. Wenn Sie beispielsweise gerade den 37. Task eines Playbooks entwickeln, kann es recht lästig sein, bei einem Testlauf immer die ersten 36 Tasks abwarten zu müssen. Eine Möglichkeit in Ansible ist, Tasks optional mit *Tags* zu versehen, wie es das folgende Beispiel aus Listing 28.28 zeigt:

```

- hosts: localhost
 gather_facts: no

 tasks:
 - name: Sag hallo
 debug: msg="Hallo Ansible!"
 tags: hallo

 - name: Eine weitere Meldung
 debug: msg="Alles soweit gut hier"
 tags: [test, meldung]

 - name: Das wars
 debug: msg="Fertig"
 tags: always
```

**Listing 28.28** »tags.yml«: ein Beispiel für den Einsatz von Tags

Beim zweiten Task sehen Sie, dass auch Listen von Tags möglich sind. Das spezielle Tag *always* bewirkt, dass der entsprechende Task immer ausgeführt wird (außer er wird explizit geskippt). Denkbare Aufrufe wären jetzt z.B.:

*Alle mit "hallo" markierten Tasks:*

```
$ ansible-playbook tags.yml -t hallo
```

*Alle mit "hallo" oder "test" markierten Tasks:*

```
$ ansible-playbook tags.yml -t hallo,test
```

*Alle Tasks außer die mit "meldung" markierten:*

```
$ ansible-playbook tags.yml --skip-tags meldung
```

Später kann das alles etwas kniffliger werden, wenn Sie Tasks entwickeln, die aufeinander aufbauen. Wenn z. B. ein früherer Task eine Variable erzeugt, die von einem späteren Task verwendet wird, dann können Sie logischerweise den späteren Task nicht losgelöst alleine verwenden.

### Wichtig

Wenn Sie mit Tags arbeiten, achten Sie stets darauf, dass auch *alle* Tasks, die Sie benötigen, entsprechend getaggt sind!



Sie haben bereits das spezielle Tag `always` gesehen; Tabelle 28.3 zeigt alle zur Verfügung stehenden »Spezialitäten«:

| Tag bzw. Aufruf                                 | Bedeutung                                                                |
|-------------------------------------------------|--------------------------------------------------------------------------|
| Tag: <code>always</code>                        | Der Task wird immer ausgeführt (außer wenn er explizit geskippt wurde).  |
| Tag: <code>never</code>                         | Der Task wird nie ausgeführt (außer wenn er explizit angefordert wurde). |
| <code>ansible-playbook [...] -t tagged</code>   | Nur Tasks ausführen, die wenigstens ein Tag haben                        |
| <code>ansible-playbook [...] -t untagged</code> | Nur Tasks ausführen, die kein Tag haben                                  |

**Tabelle 28.3** Spezielle Tags und Aufrufe

### 28.5.6 Das Kommando »ansible-playbook«

Das Kommando `ansible-playbook` ist bei der Arbeit mit Ansible sicher das zentrale Werkzeug (jedenfalls solange Sie die Kommandozeile nutzen).

Wie jedes Linux-Kommando bietet es eine Vielfalt von Optionen; wir geben nun zur Referenz einmal eine Übersicht der wichtigsten Möglichkeiten. `ansible-playbook` wird grundsätzlich wie folgt aufgerufen:

```
ansible-playbook [OPTIONEN] playbook.yml [playbook2.yml ...]
```

Ja, Sie können `ansible-playbook` auch mit mehreren Playbooks aufrufen, was aber in der Praxis eher selten vorkommt.

Dabei werden alle enthaltenen Plays in der entsprechenden Reihenfolge ausgeführt, und spätere Plays können auch auf Ergebnisse früherer Plays zugreifen. Tabelle 28.4 gibt eine Übersicht der nützlichsten Optionen.

| Option                         | Bedeutung                                                         |
|--------------------------------|-------------------------------------------------------------------|
| -l <HOST_OR_GROUP>             | Aktionen auf gewisse Hosts beschränken.                           |
| --list-hosts                   | Ausgabe der Hosts, die vom Aufruf betroffen sind                  |
| --list-tasks                   | Übersicht über die Tasks eines Playbooks                          |
| --start-at-task <NAME>         | Starte mit Task <NAME>. Joker wie »*« sind möglich.               |
| -i <INVENTORY_FILE>            | Alternatives Inventory; siehe Abschnitt 28.2.7.                   |
| -v                             | Verbose-Mode (bis zu 4 Wiederholungen möglich)                    |
| -f <NUM>                       | Anzahl der Forks; siehe Abschnitt 28.3.4                          |
| --syntax-check                 | syntaktische Überprüfung des Playbooks                            |
| --check                        | Simulierter Lauf, »Dry Run«; siehe Abschnitt 28.5.7               |
| --diff                         | Dateiänderungen anzeigen, oft mit --check; siehe Abschnitt 28.5.7 |
| --step                         | Schritt-für-Schritt-Abarbeitung                                   |
| -t <TAGS>                      | Nur markierte Tasks ausführen; siehe Abschnitt 28.5.5.            |
| -e <VAR=WERT> / -e @<file.yml> | Parametrisierung; siehe Abschnitt 28.6.1.                         |

**Tabelle 28.4** »ansible-playbook«: die wichtigsten Optionen

Das sollte für die meisten Anwendungen ausreichen. Rufen Sie bitte, um die komplette Übersicht zu erhalten, den Befehl `ansible-playbook -h` auf.

### 28.5.7 Eine exemplarische Apache-Installation

Wenden wir uns nun erstmalig einer anspruchsvolleren, aber dennoch überschaubaren Aufgabe zu: einer Apache-Installation mit kleinen Anpassungen, nämlich der Startseite und der Konfiguration. Das Ziel soll hier sein, gewisse typische Vorgehensweisen zu demonstrieren, und nicht etwa, ein komplettes Beispiel für eine Real-Life-Webserverinstallation zu liefern.

Außerdem wollen wir uns zunächst einmal auf Target Hosts aus der Debian-Familie beschränken, da der Apache-Server in unseren Testdistributionen doch sehr unterschiedlich gehandhabt wird. Diese Unterschiede (elegant) in den Griff zu bekommen, wird eine Aufgabe für später sein. Nennen Sie Ihr Playbook z. B. `apache.yml`. Wir werden uns Task für Task vorarbeiten; am Ende finden Sie das komplette Playbook am Stück.

## Schritt für Schritt

Zu Beginn klären wir die beteiligten Hosts und leiten die Tasks-Sektion ein:

```

- hosts: [debian, ubuntu]

 tasks:
```

Sie sehen hier die zwei Zielhosts, dargestellt mit der Inline-Notation einer YAML-Liste, die ein wenig kompakter daherkommt.

Beginnen wir mit dem ersten Task. Auf Debian-artigen Systemen sollte man stets mit aktuellen Paketlisten arbeiten, wenn man Installationen oder Upgrades durchführt. Debian-Anwender kennen dafür die Kommandos `apt update` oder `apt-get update`; im Ansible-Kontext erledigen Sie das mit dem `apt`-Modul. Um die Idempotenz zu ermöglichen bzw. um ständige unnötige Aktualisierungen zu vermeiden, können Sie mit dem Parameter `cache_valid_time` einen Zeitraum in Sekunden angeben, in dem der Apt-Cache als »frisch« betrachtet wird:

```
- name: Paketlisten aktualisieren
 apt:
 update_cache: yes
 cache_valid_time: 3600
```

Die eigentliche Installation erledigen Sie ebenfalls mit dem `apt`-Modul:

```
- name: Apache-Paket installieren
 apt:
 name: apache2
```

Der Start des Dienstes und die Integration in den Bootprozess sind auf Debian-Systemen eigentlich nicht nötig, weil die Postinstall-Routinen des Paketes das schon erledigen. Aber Sie werden sich ja gegebenenfalls auch recht bald um Nicht-Debian-Systeme kümmern, und da benötigen Sie diesen Schritt sehr wohl:

```
- name: Dienst starten und in Bootprozess integrieren
 service:
 name: apache2
 state: started
 enabled: yes
```

Schließlich tauschen Sie die Standard-Begrüßungsseite des Apache noch durch eine eigene Version aus. Eine Möglichkeit zum Hochladen von Dateien ist das `copy`-Modul. Typischerweise würden Sie mit dem Parameter `src` eine Quelldatei auf dem Control Host angeben, aber wir verwenden hier einmal aus Gründen der Übersichtlichkeit die direkte Inline-Notation mit dem Parameter `content`. Beachten Sie insbesondere den mit »|« eingeleiteten YAML-Block – eine sehr elegante Möglichkeit, um kleinere Textpassagen unterzubringen:

```
- name: Minimale Startseite einrichten
 copy:
 dest: /var/www/html/index.html
 mode: 0644
 content: |
 <!doctype html>
 <html>
 <head>
 <meta charset="utf-8">
 <title>Test</title>
 </head>
 <body>
 <h1>Willkommen auf unserer Homepage!</h1>
 </body>
 </html>
```

### Das komplette Playbook

Das komplette Playbook sehen Sie in Listing 28.29:

```

- hosts: [debian, ubuntu]

tasks:
 - name: Paketlisten aktualisieren
 apt:
 update_cache: yes
 cache_valid_time: 3600

 - name: Apache-Paket installieren
 apt:
 name: apache2

 - name: Dienst starten und in Bootprozess integrieren
 service:
 name: apache2
 state: started
 enabled: yes

 - name: Minimale Startseite einrichten
 copy:
 dest: /var/www/html/index.html
 mode: 0644
 content: |
```



```

<!doctype html>
<html>
 <head>
 <meta charset="utf-8">
 <title>Test</title>
 </head>
 <body>
 <h1>Willkommen auf unserer Homepage!</h1>
 </body>
</html>

```

**Listing 28.29** »apache.yml«: erstes einfaches Playbook zur Apache-Installation

Nach dem Aufruf mit `ansible-playbook apache.yml` sollten Sie im Browser die Startseiten bewundern können. Wenn Ihr Desktop-System die Namen der Linux-Maschinen nicht kennt, müssen Sie hier wohl mit IP-Adressen arbeiten, also z. B.:

```
http://192.168.150.10
```

Für ernsthaftere Arbeiten im HTTP-Umfeld ist auch das Tool `curl` eine große Hilfe; installieren Sie es bitte einmal aus dem gleichnamigen Paket, falls es auf Ihrem System noch nicht zur Verfügung steht. Damit können Sie dann blitzschnell und ganz ohne Browser den Content über HTTP abrufen (sogar auf das Präfix `http://` könnten Sie noch verzichten):

```

$ curl http://debian
<!doctype html>
[...]
 <h1>Willkommen auf unserer Homepage!</h1>
[...]

```

### »--start-at-task«, »--check«, »--diff«

Lassen Sie uns noch einen Blick auf drei nützliche Optionen für `ansible-playbook` werfen. Angenommen, Sie entwickeln gerade den letzten Task, der den Inhalt der Webseite hochlädt. Selbst bei unserem kleinen Playbook ist es dann schon fast lästig, bei jedem Lauf alle vorangehenden Tasks abwarten zu müssen. Hier kann die Option `--start-at-task` hilfreich sein, mit der man den Einstieg ins Play selbst festlegen kann. Sie müssen dabei den *Namen* des Tasks angeben, aber der Joker `»*«` ist dankenswerterweise erlaubt, etwa:

```
$ ansible-playbook apache.yml --start-at-task "Minimale Start*"
```

```

PLAY [debian,ubuntu] *****

TASK [Gathering Facts] *****
ok: [ubuntu]
ok: [debian]

```

```
TASK [Minimale Startseite einrichten] *****
ok: [debian]
ok: [ubuntu]
```

```
PLAY RECAP *****
debian : ok=2 changed=0 unreachable=0 failed=0
ubuntu : ok=2 changed=0 unreachable=0 failed=0
```

Die Idempotenz an dieser Stelle ist Ihnen sicher auch schon aufgefallen: Wenn Sie keine inhaltliche Änderung der Startseite durchführen, ist es für Ansible auch kein Change. Falls Sie jedoch eine Änderung durchführen, gibt es die interessante Möglichkeit eines Dry-Runs nebst Ausgabe der Unterschiede. Das heißt, Ansible kann Ihnen sagen, was es *tun würde*, ohne wirklich etwas zu tun. Angenommen, wir ändern die Überschrift der Startseite:

```
$ ansible-playbook apache.yml --check --diff
[...]
--- before: /var/www/html/index.html
+++ after: /home/ansible/.ansible/tmp/ansible-local-9647f_m3pf8v/tmp53n38lny
@@ -5,6 +5,6 @@
 <title>Test</title>
 </head>
 <body>
- <h1>Willkommen auf unserer Homepage!</h1>
+ <h1>Willkommen auf unserer Seite!</h1>
 </body>
 </html>

changed: [debian]
[...]
```

Man sieht, wie sich in der `<h1>`-Zeile das Wort »Homepage« in »Seite« ändern würde. Auf Ihrem System ist die Ausgabe natürlich bunt und damit noch viel besser verständlich.

### 28.5.8 Handler: Tasks nur bei Changes durchführen

Wir wollen nun unser Apache-Playbook noch etwas erweitern: Der Apache-Server soll eine kleine Konfigurationsanpassung in Form einer Plug-in-Konfigurationsdatei bekommen.

#### Schritt für Schritt

Nehmen wir exemplarisch ein Redirect (dafür muss man in aller Regel kein weiteres Apache-Modul einbinden):

```
- name: Plugin-Config hochladen
 copy:
```

```

dest: /etc/apache2/conf-available/redirect.conf
content: |
 Redirect /go http://www.google.de

```

Auf Debian-Systemen reicht das aber nicht; die neue Konfiguration muss auch eingebunden werden. Dazu steht das Debian-Kommando `a2enconf` zur Verfügung, das im benachbarten *conf-enabled*-Verzeichnis einen symbolischen Link anlegt. Das machen wir uns auch sogleich mit dem `create`-Parameter zunutze, um für Idempotenz zu sorgen:

```

- name: Config aktivieren
 command:
 cmd: a2enconf redirect
 creates: /etc/apache2/conf-enabled/redirect.conf

```

Was passiert hier? Das auszuführende Kommando ist `a2enconf redirect` und wird durch `cmd` definiert. Wenn das bereits alles wäre, würde Ansible dieses Kommando bei jedem Lauf ausführen und stets einen Change berichten, weil Commands per Default nie idempotent sind. Das wäre in diesem Fall nicht einmal schlimm – nur unschön. Um jetzt noch die Idempotenz zu erreichen, können Sie mit dem Zusatz `creates` eine Datei angeben, und wenn diese bereits vorhanden ist, spart sich Ansible einfach die Ausführung des Kommandos und berichtet »ok«. Das passt in unserem Fall prima, denn das Kommando legt bei erfolgreicher Ausführung tatsächlich eine Datei an (genauer: einen symbolischen Link). In anderen Situationen muss man mit `creates` vielleicht kritischer umgehen, denn es geht dabei immer nur um das *Vorhandensein* einer Datei und nicht um die Frage, *ob auch das Richtige drinsteht*.

Zu guter Letzt wird natürlich noch ein Reload des Apache-Dienstes fällig:

```

- name: reload apache
 service:
 name: apache2
 state: reloaded

```

Lassen Sie das Playbook mit den neuen Schritten ablaufen. Danach testen Sie im Browser oder wieder einfach mit `curl`:

```

$ curl http://debian/go
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved here.</p>
<hr>
<address>Apache/2.4.38 (Debian) Server at debian Port 80</address>
</body></html>

```

So weit ist alles prima. Etwas störend ist nur noch die Tatsache, dass die `reloaded-` und `restarted-`Zustände des `service`-Moduls *niemals idempotent* sind. Oder einfacher gesagt: Der Reload wird immer durchgeführt, egal ob eine Konfigurationsänderung stattgefunden hat oder nicht. Und das ist die Stelle, wo die *Handler* ins Spiel kommen.

## Handler

Eine elegante Lösung des Problems unnötiger Reloads/Restarts sind *Handler*, die nur dann aufgerufen werden, wenn auch wirklich eine Änderung stattgefunden hat.

Dazu müssen Sie in dem Task, der die Änderung gegebenenfalls durchführt, den Handler mit `notify` benachrichtigen:

```
- name: Plugin-Config hochladen
 copy:
 dest: /etc/apache2/conf-available/redirect.conf
 content: |
 Redirect /go http://www.google.de
 notify: reload apache
```

Beachten Sie dabei wieder die Einrückung: `notify` steht auf der Ebene des Tasks, es ist *kein* `copy`-Parameter!

Der eigentliche Reload-Task wandert nun in eine neue Play-Sektion namens `handlers`, die auf derselben Ebene wie `tasks` eröffnet wird:

```
handlers:
- name: reload apache
 service:
 name: apache2
 state: reloaded
```

Wenn Sie das Ganze nun testen, sollte alles wie gewünscht funktionieren: keine Änderung – kein Reload. Wenn Sie aber das `Redirect` testweise irgendwie abändern, dann erhalten Sie folgendes Ergebnis:

```
$ ansible-playbook apache.yml
[...]
TASK [Plugin-Config hochladen] *****
changed: [debian]
changed: [ubuntu]
[...]

RUNNING HANDLER [reload apache] *****
changed: [debian]
changed: [ubuntu]
```

```
PLAY RECAP *****
debian : ok=8 changed=2 unreachable=0 failed=0
ubuntu : ok=8 changed=2 unreachable=0 failed=0
[...]
```

### Erläuterungen

- ▶ Ein Handler wird im Task durch ein `notify` benachrichtigt. Er wird über seinen Namen angesprochen. Falls mehrere Handler benachrichtigt werden müssen, können Sie an `notify` auch eine Liste übergeben.
- ▶ Die Handler laufen (wenn überhaupt) stets am Ende des Plays.
- ▶ Die Handler laufen in genau der Reihenfolge, in der sie unter `handlers` spezifiziert werden.
- ▶ **Wenn ein Play fehlschlägt, laufen die Handler nicht!**

Der letzte Punkt ist beachtenswert (und eigentlich *nicht* wünschenswert). Seit Version 1.9.1 können Sie dieses Standardverhalten ändern, und zwar gleich auf drei verschiedene Weisen:

1. mit dem Kommandozeilenschalter `--force-handlers`
2. mit der Einstellung `force_handlers: true` im Playbook
3. (Empfohlen:) mit der Einstellung `force_handlers = true` in der *ansible.cfg*:

```
[defaults]
force_handlers = true
```

**Listing 28.30** »ansible.cfg«: Ausschnitt

### Handler früher aufrufen

Recht häufig besteht die Anforderung, gewisse Handler nicht erst am Ende des Plays aufzurufen, sondern schon früher. Bei Bedarf können Sie den Aufruf aller bislang benachrichtigten Handler wie folgt erzwingen:

```
[... Tasks ...]

- meta: flush_handlers

[... Tasks ...]
```

**Listing 28.31** Handler zu einem früheren Zeitpunkt aufrufen

### Entkopplung vom Namen

Es ist auch möglich, mit dem Parameter `listen` die Notification vom Handler-Namen zu entkoppeln. Dabei ist besonders praktisch, dass verschiedene Handler damit auch auf dieselbe Notification hören können. Hier folgt sinngemäß ein Beispiel aus der Ansible-Dokumentation:

```
tasks:
 - name: restart everything
 command: echo "this task will restart the web services"
 notify: "restart web services"

handlers:
 - name: restart memcached
 service: name=memcached state=restarted
 listen: "restart web services"

 - name: restart apache
 service: name=apache state=restarted
 listen: "restart web services"
```

## 28.6 Playbooks und Tasks: fortgeschrittene Methoden

Das zentrale Thema dieses Abschnitts ist der Umgang mit *Variablen*. So gut wie kein reales Playbook kommt ohne sie aus, denn erst mit Variablen wird es möglich, Einstellungen bzw. Werte zu zentralisieren oder Unterschiede zwischen Zielsystemen effizient auszugleichen. Weitere Punkte sind u. a. der Umgang mit Jinja-Templates und Kontrollstrukturen wie Bedingungen und Schleifen.

### 28.6.1 Variablen

Ansible bietet die Möglichkeit, in Tasks auf Variablen zuzugreifen. Dazu muss eine solche Variable zuvor an anderer Stelle definiert worden sein. Es gibt unzählige Möglichkeiten, um dies zu tun (etwas genauer: über 20). In der Folge schauen wir uns einige dieser Möglichkeiten an.

#### Play Vars

Eine der vielen Möglichkeiten zur Variablendefinition ist eine *vars*-Sektion im Play. Im folgenden Beispiel aus Listing 28.32 sehen Sie, wie zwei Variablen definiert werden, deren Inhalt dann im einzigen Task mit dem *debug*-Modul ausgegeben wird:

```

- hosts: localhost
 gather_facts: false

 vars:
 farbe: blau
 zahl: 42
```

```

tasks:
 - debug:
 msg: "Farbe: {{farbe}}, Zahl: {{zahl}}"

```

**Listing 28.32** »vars1.yml«: erstes Beispiel für eine Variablendefinition im Playbook

```

$ ansible-playbook vars1.yml
[...]
ok: [localhost] => {
 "msg": "Farbe: blau, Zahl: 42"
}
[...]

```

So weit gibt es keine Überraschungen. Die für den Variablenzugriff benötigte Syntax mit den doppelten geschweiften Klammern '`{{...}}`' nehmen wir einfach mal so hin – sie ist vielleicht ungewohnt, aber durchaus logisch.

### Extra Vars

Eine weitere Möglichkeit zur Variablendefinition ist die Übergabe per Kommandozeile (Ansible nennt dies *Extra Vars*). Zur Demonstration nutzen wir einmal dasselbe Beispiel-Playbook aus dem letzten Abschnitt:

```

$ ansible-playbook vars1.yml -e zahl=100
[...]
ok: [localhost] => {
 "msg": "Farbe: blau, Zahl: 100"
}
[...]

```

Die Übergabe per Kommandozeile übertrumpft die Definition im Play – auch keine große Überraschung. Den Schalter `-e` können Sie bei Bedarf natürlich beliebig oft wiederholen:

```

$ ansible-playbook vars1.yml -e farbe=rot -e zahl=100

```

### Präzedenzen

Wenn Sie dieselbe Variable an unterschiedlichen Stellen definieren, dann wird eine davon »gewinnen«. Beispielsweise ist, wie wir gesehen haben, eine Extra-Vars-Variable stärker als eine Play-Variable. Natürlich sind die Präzedenzregeln klar definiert; die ausführliche Rangliste der Prioritäten finden Sie unter [http://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_variables.html](http://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html) im Abschnitt »*Variable Precedence: Where Should I Put A Variable?*«. Die Ordnung der Liste geht von *schwach nach stark*, d.h.: Was weiter unten steht, gewinnt.

1. command line values (eg `"-u user"`)
2. role defaults
3. inventory file or script group vars

4. inventory group\_vars/all
5. playbook group\_vars/all
6. inventory group\_vars/\*
7. playbook group\_vars/\*
8. inventory file or script host vars
9. inventory host\_vars/\*
10. playbook host\_vars/\*
11. host facts / cached set\_facts
12. play vars
13. play vars\_prompt
14. play vars\_files
15. role vars (defined in role/vars/main.yml)
16. block vars (only for tasks in block)
17. task vars (only for the task)
18. include\_vars
19. set\_facts / registered vars
20. role (and include\_role) params
21. include params
22. extra vars (always win precedence)

Die Play Vars finden Sie an Position 12 und die Extra Vars an Position 22. Damit können Sie jetzt wirklich sicher sein: Extra Vars auf der Kommandozeile gewinnen immer!

### »set\_fact«

Mitunter sollen Variablen erst während der Laufzeit eines Plays definiert werden. Mittels `set_fact` können Sie jederzeit bei Bedarf Variablen ins Spiel bringen bzw. vorhandene Variablen mit einem neuen Wert versehen:

```

- hosts: localhost
 gather_facts: false

 tasks:
 - name: Variablen definieren
 set_fact:
 farbe: blau
 zahl: 42

 - debug:
 msg: "Farbe: {{farbe}}, Zahl: {{zahl}}"
```

**Listing 28.33** »set\_fact.yml«: Variablendefinition mit »set\_fact«

In der Präzedenzliste finden Sie `set_fact` auf Position 19 (es ist also ziemlich stark).



**Anmerkung**

Etwas Erklärungsbedarf hat vielleicht noch der ungewöhnliche Name `set_fact`. Wäre etwas wie »`set_variable`« nicht plausibler gewesen?

Fakten bzw. Facts, die Sie in Kürze in Abschnitt 28.6.3 kennenlernen werden, sind in Ansible *host-spezifische Variablen*. Und dasselbe trifft auf Variablen zu, die Sie mittels `set_fact` definieren: Sie sind ab dem Zeitpunkt der Definition nur für den Host sichtbar, der den `set_fact`-Task gerade ausgeführt hat.

Man hat diesen Namen also wohl gewählt, um die konzeptuelle Verwandtschaft zu den Facts deutlich zu machen.

**»group\_vars«**

Um Unterschiede zwischen Systemen zu handhaben, brauchen wir Methoden, mit denen eine Variable je nach System einen anderen Wert zugewiesen bekommt. Wir nehmen dazu erst einmal alle Hosts an Bord, und sämtliche Variablendefinitionen lassen wir heraus:

```

- hosts: all
 gather_facts: false

 tasks:
 - debug:
 msg: "Farbe: {{farbe}}, Zahl: {{zahl}}"
```

**Listing 28.34** »vars2.yml«: ein von externer Parametrisierung abhängiges Playbook

Damit können wir nebenbei auch direkt verifizieren, dass der Zugriff auf eine undefinierte Variable mit einem fatalen Laufzeitfehler endet:

```
$ ansible-playbook vars2.yml
```

```
TASK [debug] *****
fatal: [ubuntu]: FAILED! => {"msg": "The task includes an option with an
undefined variable. The error was: 'farbe' is undefined[...]
```

[... dasselbe für alle anderen Hosts ...]

Welche Möglichkeiten gibt es nun, Variablen außerhalb eines Playbooks zur Verfügung zu stellen (abgesehen von den Extra Vars)? Ansible sucht u. a. in einem Verzeichnis mit dem Namen `group_vars`, und zwar

- ▶ parallel zum benutzten Inventory oder
- ▶ parallel zum benutzten Playbook.

In diesem Verzeichnis erwartet Ansible dann wahlweise

- ▶ *entweder* eine Datei `<GRUPPENNAME>.yml`
- ▶ *oder* ein Verzeichnis `<GRUPPENNAME>/` mit beliebig benannten `*.yml`-Dateien.

Wenn Sie es versuchen möchten, dann legen Sie nun einen Ordner `group_vars` an. Ob Sie ihn parallel zum Inventory oder im Order `playbooks/` anlegen, ist eine fachliche Frage: Sind die Parameter eher an die Hosts gebunden, oder sind sie eher abhängig von gewissen Playbooks? In unserem Spielbeispiel ist das natürlich egal, testen Sie einmal die erste Variante:

```
$ mkdir ~/ansible/projects/start/group_vars
```

Lassen Sie uns nun die Tatsache nutzen, dass die Gruppe `all` jeden Host enthält, und platzieren Sie in `group_vars/` die folgende Datei `all.yml`, die somit Default-Werte für alle Hosts zur Verfügung stellt:

```

farbe: LightGray
zahl: 50
```

**Listing 28.35** »`group_vars/all.yml`«: Default-Werte für alle Hosts

Und schon sollte unser Playbook auch erfolgreich laufen können:

```
$ ansible-playbook vars2.yml
[...]
ok: [debian] => {
 "msg": "Farbe: LightGray, Zahl: 50"
}
[...]
```

Damit haben wir zwar immer noch keine individuelle Parametrisierung, aber nun könnten wir ja problemlos weitere Dateien für speziellere Gruppen bereitstellen. Angenommen, Sie möchten alle SUSE-Hosts »grün färben«. Dazu könnten Sie im Inventory eine Gruppe `suse_hosts` definieren:

```
[suse_hosts]
suse
...
```

**Listing 28.36** »`inventory`«: Inventory-Abschnitt mit zusätzlicher Gruppe

Dadurch würde Ansible auch die folgende Datei `group_vars/suse_hosts.yml` als Parameterdatei beachten und einlesen:

```

farbe: Green
```

**Listing 28.37** »`group_vars/suse_hosts.yml`«: gesonderte Einstellung für SUSE-Hosts

Und damit ergibt sich:

```
$ ansible-playbook vars2.yml
[...]
ok: [ubuntu] => {
 "msg": "Farbe: LightGray, Zahl: 50"
}
ok: [suse] => {
 "msg": "Farbe: Green, Zahl: 50"
}
[...]
```

Wie gehabt, möchten wir auch bei dieser Methode den Präzedenzrang benennen: Mit unserer Datei `group_vars/all.yml` liegen wir auf Rang 4 (sehr schwach) und mit einer spezielleren Datei `group_vars/*.yml` immerhin auf Rang 6 (etwas stärker).

### »host\_vars«

Vermutlich können Sie sich schon denken, wie es hier weitergeht: Ansible unterstützt auch individuelle Parametrisierung bis auf die Ebene einzelner Hosts. Alle Konzepte funktionieren analog zu `group_vars`, aber wir halten trotzdem noch mal fest:

Ansible sucht auch in einem Verzeichnis mit dem Namen `host_vars`, und zwar

- ▶ parallel zum benutzten Inventory oder
- ▶ parallel zum benutzten Playbook.

In diesem Verzeichnis erwartet Ansible dann wahlweise

- ▶ *entweder* eine Datei `<HOSTNAME>.yml`
- ▶ *oder* ein Verzeichnis `<HOSTNAME>/` mit beliebig benannten `*.yml`-Dateien darin.

Individuelle Hostparameter müssen stärker sein als Gruppenparameter, und ein kurzer Blick in die Präzedenzliste bestätigt dies auch: Wir liegen mit dieser Methode auf Rang 9.

### Zugriffe auf komplexe Strukturen

Variablen können nicht nur simple Einzelwerte enthalten, sondern auch komplexe verschachtelte Strukturen, wie das folgende Beispiel zeigt:

```

- hosts: all
 gather_facts: no

 vars:
 motd:
 gruss: Hallo lieber Besucher!
```

```
 spruch:
 - Früher Vogel fängt den Wurm.
 - Was Hänschen nicht lernt, lernt Hans nimmermehr.
 lotto: [2, 8, 17, 33, 34, 42]

tasks:
 - name: /etc/motd hochladen
 copy:
 dest: /etc/motd
 content: |
 {{ motd.gruss }}

 Spruch des Tages: {{ motd.spruch[0] }}

 Lottozahlen vom Wochenende: {{ motd.lotto | join(', ') }}
```

**Listing 28.38** »vars-complex.yml«: Arbeiten mit komplex strukturierten Variablen

Hierzu einige Erklärungen:

- ▶ Auf Werte in verschachtelten Maps greifen Sie mit folgender Notation zu:  
hash.key1.key2...  
Alternativ steht die Syntax  
hash['key1']['key2'][...]  
zur Verfügung.
- ▶ Auf Listenwerte greifen Sie mit folgender Notation zu:  
liste[POS]  
Die Indizierung beginnt wie üblich bei 0.
- ▶ Um die Listenelemente wieder zu einem Text »zusammenzukleben«, steht Ihnen der Jinja2-Filter `join()` zur Verfügung (dazu folgt mehr in Abschnitt 28.6.5).

## 28.6.2 Registrierte Variablen

Gerade in der Anfangsphase, wenn man beginnt, sich mit Ansible vertraut zu machen, wird man des Öfteren Linux-Kommandos mit `command` oder `shell` aufrufen. Im Nachhinein wäre dabei ein spezielles Modul meist die bessere Wahl gewesen: das `file`-Modul statt eines `mkdir`-Aufrufs, das `user`-Modul statt eines `useradd`-Aufrufs, das `synchronize`-Modul statt eines `rsync`-Aufrufs. Aber erstens muss man das alles erst mal wissen, und zweitens gibt es auch nicht für alles ein Modul. Wir sollten uns also einmal anschauen, wie sich `command`- und `shell`-Aufrufe im Playbook-Kontext verhalten. (Als Ad-hoc-Aufrufe kennen Sie diese schon; siehe Abschnitt 28.3.1.) Beginnen wir wieder mit einem simplen Beispiel:

```

- hosts: debian
 gather_facts: no

 tasks:
 - name: Platzbedarf der Root-Partition ermitteln
 command: df -h /

```

**Listing 28.39** »command1.yml«: Aufruf eines Linux-Kommandos

```
$ ansible-playbook command1.yml
```

```

PLAY [debian] *****

TASK [Platzbedarf der Root-Partition ermitteln] *****
changed: [debian]

PLAY RECAP *****
debian : ok=1 changed=1 unreachable=0 failed=0

```

28

Es fällt sofort auf, dass man – im Gegensatz zu einem Ad-hoc-Aufruf – nichts sieht. Und auf den zweiten Blick sehen Sie noch, dass Ansible einen Change berichtet (obwohl wir ziemlich sicher sind, dass ein `df -h` keine Änderungen am Zielsystem verursachen sollte).

Letzteres ist immerhin einfach erklärt: Ansible nimmt keine tiefe Analyse der aufzurufenden Linux-Kommandos vor und weiß deshalb nicht, was wir hier tun. Per Default geht es daher bei einem solchen Task stets von einer Änderung aus. Sie können jedoch mit dem Task-Attribut `changed_when` diese Beurteilung selbst definieren. Sobald Sie Bedingungen in Ansible kennengelernt haben (in Abschnitt 28.6.4), stehen Ihnen hier alle Möglichkeiten offen; in unserem aktuellen Fall hilft aber bereits eine grundsätzliche Verneinung mittels `changed_when: false`. Ein `df`-Kommando verursacht schließlich nie eine Änderung am System:

```

- hosts: debian
 gather_facts: no

 tasks:
 - name: Platzbedarf der Root-Partition ermitteln
 command: df -h /
 changed_when: false

```

**Listing 28.40** »command2.yml«: Aufruf eines Linux-Kommandos ohne Change

Es bleibt aber noch das Problem, dass Sie die Ausgabe des Kommandos nicht sehen. Zunächst einmal sollten Sie sich mit der Tatsache anfreunden, dass Playbooks nicht dazu gedacht sind,

schönen Output zu produzieren. Sie sollen vielmehr Aktionen auf Zielsystemen durchführen, damit diese einen gewünschten Zielzustand erreichen; hübsche Ausgaben sind dabei irrelevant. Deswegen werden auch alle Standardausgaben von `command-` oder `shell-`Aufrufen unterdrückt.

Durch einen kleinen Umweg kommen wir aber doch an sie heran (und noch an viele weitere Informationen): Ansible bietet die Möglichkeit, alle Informationen zum Ablauf eines Tasks strukturiert in einer Variablen zu speichern (Schlüsselwort `register`). Auf diese können Sie dann in späteren Tasks zugreifen:

```

- hosts: debian
 gather_facts: no

 tasks:
 - name: Platzbedarf der Root-Partition ermitteln
 command: df -h /
 changed_when: false
 register: df_cmd

 - debug:
 msg: '{{df_cmd.stdout_lines}}'
```

**Listing 28.41** »`command3.yml`«: Aufruf eines Linux-Kommandos mit anschließender Ausgabe

```
$ ansible-playbook command3.yml
```

```
PLAY [debian] *****

TASK [Platzbedarf der Root-Partition ermitteln] *****
ok: [debian]

TASK [debug] *****
ok: [debian] => {
 "msg": [
 "Filesystem Size Used Avail Use% Mounted on",
 "/dev/mapper/debian--10--vg-root 62G 1.2G 58G 2% /"
]
}

PLAY RECAP *****
debian : ok=2 changed=0 unreachable=0 failed=0
```

Wir möchten zunächst ganz deutlich festhalten, wie registrierte Variablen technisch einzuordnen sind:

**Wichtig**

Registrierte Variablen sind technisch dasselbe wie `set_fact`-Variablen: Sie sind *host-spezifisch*, also nur für die Hosts sichtbar, die den Task gerade ausgeführt haben. Und sie haben exakt dieselbe Präzedenz.

Eine registrierte Variable hat – je nach Art des Tasks – verschiedenste Felder bzw. Attribute. Wir haben hier auf das Feld `stdout_lines` zugegriffen. Es enthält eine Liste der einzelnen Ausgabezeilen und kommt damit der gewohnten Ausgabe relativ nahe.

Tabelle 28.5 zeigt, welche Felder insgesamt zum Vorschein kämen, würden Sie sich im `debug`-Task die komplette Variable `df_cmd` ausgeben lassen:

28

| Attribute                 | Wert                                                      |
|---------------------------|-----------------------------------------------------------|
| <code>changed</code>      | Hat der Task eine Änderung am System durchgeführt?        |
| <code>failed</code>       | Ist der Task fehlgeschlagen?                              |
| <code>cmd</code>          | Kommando mit Parametern                                   |
| <code>rc</code>           | UNIX-Returncode des Kommandos                             |
| <code>start</code>        | Startzeit                                                 |
| <code>end</code>          | Endezeit                                                  |
| <code>delta</code>        | benötigte Ausführungszeit                                 |
| <code>stdout</code>       | Standardausgabe am Stück mit Newlines als <code>\n</code> |
| <code>stdout_lines</code> | Liste der einzelnen Standardausgabezeilen                 |
| <code>stderr</code>       | Standardfehlerausgabe analog zu <code>stdout</code>       |
| <code>stderr_lines</code> | Standardfehlerausgabe analog zu <code>stdout_lines</code> |

**Tabelle 28.5** Attribute eines »command«- oder »shell«-Tasks

**Tipp**

Das `debug`-Modul bietet noch eine einfachere Methode, um den Inhalt einer Variablen auszugeben. Alternativ zu

```
- debug: msg='{{name_der_variablen}}'
```



können Sie auch einfach

```
- debug: var=name_der_variablen
```

verwenden.

### 28.6.3 Facts und implizite Variablen

#### Facts

Wenn Sie in einem Playbook `gather_facts` auf `true/yes` setzen oder die Zeile gleich ganz weglassen, so sammelt Ansible zu Beginn erst einmal Informationen über den bzw. die Target Host(s). Diese stehen dann in der Datenstruktur `ansible_facts` zur Verfügung und können genau wie normale Variablen verwendet werden.

Am besten führen Sie für irgendeinen Host einmal

```
$ ansible <HOST> -m setup
```

aus und schauen sich das Ganze in Ruhe an. (Der Host `debian` aus unserer Laborumgebung produziert dabei beispielsweise über 500 Textzeilen, je nach System und Umgebung können es auch schon mal über 700 sein. Diese hier abzdrukken, ersparen wir uns einfach mal.)

#### Zugriff auf Facts

Die Ausgabe des `setup`-Moduls ist leider etwas irreführend. Schauen Sie sich einmal den oberen Bereich an, so finden Sie dort beispielsweise den Schlüssel `ansible_architecture` mit einem einfachen Wert wie `x86_64`:

```
$ ansible -m setup debian
debian | SUCCESS => {
 "ansible_facts": {
 "ansible_all_ipv4_addresses": [
 "10.0.2.15",
 "192.168.57.10"
],
 "ansible_all_ipv6_addresses": [
 "fe80::a00:27ff:fe8b:e4c6",
 "fe80::a00:27ff:fe66:70d3"
],
 "ansible_apparmor": {
 "status": "enabled"
 },
 "ansible_architecture": "x86_64",
```

```
[...]
```



Wenn Sie diesen Wert nun in einem Playbook verwenden möchten, würden Sie nach allem, was Sie in Abschnitt 28.6.1 erfahren haben, sicher etwas schreiben wie:

- debug: var=ansible\_facts.ansible\_architecture
- debug: var=ansible\_facts['ansible\_architecture']

Nur leider bekommen Sie mit beiden Varianten nur »Undefiniert« als Wert zurück. Erfolgreich wären Sie hingegen mit:

- debug: var=ansible\_facts.architecture
- # oder (nur andere Notation)
- debug: var=ansible\_facts['architecture']
- # oder(!)
- debug: var=ansible\_architecture

Halten wir daher fest:

- ▶ Wenn Sie über die Map `ansible_facts` auf Fakten zugreifen, so müssen Sie die Schlüssel der ersten Ebene stets *ohne* das Präfix `ansible_` verwenden.
- ▶ Die Schlüssel der ersten Ebene stehen außerdem auch als Direktvariable *mit* dem Präfix `ansible_` zur Verfügung.

Ob man Letzteres will oder nicht ist konfigurierbar, und zwar mit der Einstellung `inject_facts_as_vars`. Diese wurde mit Ansible 2.5 eingeführt und steht defaultmäßig auf `true`.

### Cachen von Facts

Das Zusammensuchen der Facts auf einem Host kann sich durchaus zeitintensiv oder performance-lastig gestalten. Es gibt aber die Möglichkeit, das Ergebnis für einen definierten Zeitraum zu cachen – in einfachen JSON- oder YAML-Dateien oder in einer NoSQL-Datenbank (zumeist Redis). Mit folgender Konfiguration erreichen Sie ein Caching mit JSON-Files (was völlig ausreichend ist, solange Sie den Cache nicht zentral einrichten wollen):

```
[defaults]
gathering = smart
fact_caching = jsonfile
fact_caching_connection = ~/.ansible/fact_cache
fact_caching_timeout = 86400
```

#### Listing 28.42 »ansible.cfg«: Ausschnitt

`fact_caching_connection` und `fact_caching_timeout` sind dabei natürlich frei wählbar. Um den Cache vorzeitig zu erneuern, stehen Ihnen mehrere Möglichkeiten zur Verfügung:

- ▶ das manuelle Löschen der Cache-Dateien

- ▶ ein Aufruf des setup-Moduls als Ad-hoc-Kommando oder Playbook-Task
- ▶ der Aufruf von `ansible-playbook --flush-cache <playbook.yml>`

### Implizite Variablen

In Ansible gibt es auch *implizite Variablen*, die auf jeden Fall immer da sind. Tabelle 28.6 zeigt die wichtigsten:

| Variable                      | Bedeutung                                              |
|-------------------------------|--------------------------------------------------------|
| inventory_hostname            | der Name des Hosts im Inventory                        |
| inventory_hostname_short      | Inventory-Hostname ohne Domain                         |
| group_names                   | Liste aller Gruppen, in denen der Host Mitglied ist    |
| groups                        | Map mit allen Inventory-Gruppen und deren Inhalt       |
| hostvars                      | Hostvariablen aller Inventory-Hosts (inkl. Facts)      |
| playbook_dir                  | Verzeichnis, in dem das aktuelle Playbook liegt        |
| inventory_dir, inventory_file | Directory bzw. Datei des aktuellen Inventory-Hosts     |
| role_path, role_name          | Datei bzw. Name der aktuellen Rolle (dazu später mehr) |
| ansible_version               | Versionsinformationen zum eingesetzten Ansible         |
| ansible_check_mode            | Gibt an, ob der Check-Mode aktiviert ist oder nicht.   |

**Tabelle 28.6** Implizite Ansible-Variablen

Für eine komplette Übersicht siehe auch [http://docs.ansible.com/ansible/latest/reference\\_appendices/special\\_variables.html](http://docs.ansible.com/ansible/latest/reference_appendices/special_variables.html).

#### 28.6.4 Bedingte Ausführung mit »when«

In Ansible können Sie Tasks mit einer Bedingung verknüpfen, d. h., sie nur dann ausführen lassen, wenn eine Bedingung erfüllt ist. Dies gelingt mit dem zusätzlichen Task-Attribut `when`; dabei können Facts sowie eigene Variablen mit einbezogen werden, wie das folgende Beispiel zeigt:

```
- name: Ausgabe nur auf Debian-artigen-Systemen
 debug: msg="Hallo Debian-System!"
 when: ansible_os_family == "Debian"
```

Bedingungen können wie üblich mit `not`, `and` und `or` verknüpft werden. Natürlich sind auch runde Klammern »(...)« erlaubt, um Präzedenzen zu klären oder die Verständlichkeit zu erhöhen.

### Anmerkung

Technisch gesehen wird eine `when`-Bedingung als Jinja-Ausdruck ausgewertet, worauf wir in Abschnitt 28.6.5 zu sprechen kommen. Momentan schauen Sie sich aber am besten einfach die noch folgenden Beispiele an und stimmen uns hoffentlich zu, dass die Syntax ziemlich »plausibel« ist.



Ähnlich wie in SQL steht auch ein `in`-Operator zur Verfügung, um das Vorkommen in Listen zu prüfen:

```
when: ansible_os_family in ["Debian", "Suse"]
```

Wenn Sie einfach eine Variable prüfen oder vergleichen möchten, aber nicht klar ist, ob diese überhaupt einen definierten Wert hat, verwenden Sie Konstrukte wie:

```
when: myvar is defined
when: myvar is undefined
when: myvar is defined and myvar == "wasauchimmer"
```

Fürs Erste sollte das genügen; weitere Möglichkeiten können Sie diesen Seiten entnehmen:

- ▶ [http://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_conditionals.html](http://docs.ansible.com/ansible/latest/user_guide/playbooks_conditionals.html)
- ▶ [http://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_tests.html](http://docs.ansible.com/ansible/latest/user_guide/playbooks_tests.html)

## 28.6.5 Jinja und Templates

Sie haben bereits das `copy`-Modul kennengelernt, mit dem Sie Dateien bzw. Inhalte auf Target Hosts transferieren können. Das ist sicher eine sehr nützliche Sache, aber mitunter reicht es noch nicht, alle Zielsysteme mit derselben Version einer Datei auszustatten. An diesem Punkt kommen dann Begriffe wie *Templates* und *Jinja* ins Spiel, die wir in diesem Abschnitt beleuchten möchten.

### Begriffsklärung: Templates und Template-Engines

*Template* bedeutet »Vorlage« oder »Schablone«. Fortgeschrittene Linux-User kennen Kommandos wie

```
$ find . -name "*.sh" -exec chmod 700 {} \;
```

oder:

```
$ echo 'Hallo %NAME%! ' | sed 's/%NAME%/John/'
```

Dabei sind `chmod 700 {}` und `Hallo %NAME!` die Vorlagen, aus denen durch Ersetzen der Platzhalter ein konkretes Ergebnis gewonnen wird. Die Kommandos `find` bzw. `sed` wären dann die *Template-Engines*, also die »Maschinen«, die aus der Vorlage ein konkretes, »lebendiges« Ergebnis herstellen.

Blickt man in die Welt der Programmiersprachen, so lässt sich feststellen, dass jede prominente Sprache auch eine Template-Engine besitzt. Diese steht in den meisten Fällen als Zusatzbibliothek zur Verfügung; nicht selten hat man hier sogar mehrere Varianten zur Verfügung. In der Programmiersprache Python ist die Template-Engine Jinja (<https://jinja.palletsprojects.com>) gewissermaßen der Hoflieferant in Sachen Templates und kommt deswegen auch in Ansible zum Einsatz. Da Ansible die Version 2 dieser Bibliothek nutzt, ist genauso oft auch die Rede von *Jinja2*. Für uns sind die Begriffe Jinja und Jinja2 synonym. Jinja ist uns schon sehr oft begegnet. Immer wenn wir in einem Playbook einen Variablenzugriff realisieren, kommt Jinja zum Einsatz:

```
- debug: msg="Die Farbe ist {{farbe}}"
```

Sie denken sich jetzt möglicherweise: »Na, ist das denn so schwierig, dass man für ein simples Suchen-und-Ersetzen eine komplette Bibliothek benötigt?« Sicher nicht, wenn das schon alles wäre. Aber wir nutzen damit momentan vielleicht ein Prozent der Fähigkeiten von Jinja; etwas mehr erfahren Sie auf den folgenden Seiten.

### Jinja-Syntax: Ausgabeausdrücke, Anweisungen, Kommentare

Im Wesentlichen stellt Jinja diese drei syntaktischen Elemente zur Verfügung:

- ▶ `{{ ... }}` für Ausdrücke, die nach Auswertung in die Ausgabe übernommen werden
- ▶ `{% ... %}` für Anweisungen bzw. Kontrollstrukturen
- ▶ `{# ... #}` für Kommentare, die nicht in die Ausgabe übernommen werden

### Ausdrücke

Die einfachsten Jinja-Ausdrücke sind Zahlen (Integer oder Fließkomma) und Texte (quitiert mit einfachen oder doppelten Hochkommas). Dazu kommen noch die booleschen Ausdrücke `true` und `false`. Mit diesen Grundbausteinen können Sie dann komplexere Ausdrücke aufbauen. Hier stehen die für Programmiersprachen typischen Möglichkeiten zur Verfügung:

- ▶ **mathematische/arithmetische Ausdrücke**  
mit üblichen Rechensymbolen wie `+`, `-`, `(...)` etc.
- ▶ **Vergleiche**  
mit üblichen Symbolen wie `==`, `!=` etc.
- ▶ **logische Verknüpfungen**  
mit `and`, `or`, `not`

► **diverse weitere Operatoren**

wie z.B. in (Listenzugehörigkeit), | (Filter) oder ~ (Stringkonkatenation)

► **Listen und Maps**

[ <objekt1>, <objekt2>, ... ] bzw. { 'key1': <value1>, 'key2': <value2>, ... }

<https://jinja.palletsprojects.com/templates/#expressions> gibt die komplette Übersicht.

## Anweisungen

Zur Steuerung des Kontrollflusses stehen u.a. folgende Elemente zur Verfügung:

► **Zuweisungen mit set**

Beispiel:

```
{% set zahl = 70 %}
{% set tiere = ['Hund', 'Katze', 'Maus'] %}
```

► **Testen von Bedingungen mit if**

Beispiel:

```
{% if zahl >= 0 and zahl < 50 %}
Die Zahl ist zu klein.
{% elif zahl >= 50 and zahl <= 100 %}
Die Zahl ist groß genug.
{% else %}
Die Zahl ist nicht im gültigen Bereich.
{% endif %}
```

► **Schleifen mit for**

Beispiel:

```
{% for t in tiere %}
{{ t }}
{% endfor %}
```

<https://jinja.palletsprojects.com/templates/#list-of-control-structures> gibt die komplette Übersicht.

## Filter

Im Ansible-Alltag oft nützlich sind die Jinja-Filter; in Syntax und Semantik sind sie mit dem UNIX-Pipe-Mechanismus vergleichbar. Es folgen einige Beispiele, beginnend mit dem int-Filter, der Fließkommazahlen in ganze Zahlen konvertiert:

```
{{ (10 / 3) | int }} {# --> 3 #}
```

Der join-Filter »verklebt« Listenelemente zu einem Text mit frei wählbarem Kleber:

```
{% set tiere = ['Hund', 'Katze', 'Maus'] %}
```

```
{{ tiere | join(':') }} {# --> 'Hund:Katze:Maus' #}
```

Die upper- und lower-Filter konvertieren Text in Groß- bzw. Kleinbuchstaben:

```
{{ "Das wird gross" | upper }} {# --> "DAS WIRD GROSS" #}
{{ "DAS WIRD KLEIN" | lower }} {# --> "das wird klein" #}
```

Der default()-Filter sorgt im Falle einer undefinierten Variablen für einen Wert, mit dem gearbeitet werden kann:

```
{{ zahl | default(42) }}
```

Der replace()-Filter implementiert ein Suchen-und-Ersetzen:

```
{{ "Das ist falsch" | replace("falsch", "richtig") }}
```

Es gibt in Jinja etwa 50 dieser Filter; in der offiziellen Jinja-Dokumentation ist ihnen natürlich auch ein eigener Abschnitt gewidmet: <https://jinja.palletsprojects.com/templates/#builtin-filters>. Verschaffen Sie sich bitte dort bei Gelegenheit einmal einen groben Überblick.

Ansible definiert darüber hinaus sogar noch zahlreiche zusätzliche Filter. Es existieren Filter zum Generieren von (Passwort)-Hashes wie hash() oder password\_hash(), Filter zur Pfadnamenanalyse wie basename und dirname, Filter zur URL-Analyse (urlsplit), Filter mit regulären Ausdrücken (regex\_search(), regex\_replace()) und vieles andere mehr. Vergleichsweise oft werden Sie noch dem bool-Filter begegnen; er konvertiert textuelle Darstellungen von Wahrheitswerten wie beispielsweise "True", "FALSE", "yes" oder "0" in richtige Wahrheitswerte:

```
{% if "false" | bool %}Richtig!{% else %}Falsch!{% endif %}
```

(Lassen Sie probeweise einmal den bool-Filter weg, und Sie sehen den Unterschied im Ergebnis.)

Wir müssen auch an dieser Stelle auf die offizielle Online-Dokumentation verweisen: [http://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_filters.html](http://docs.ansible.com/ansible/latest/user_guide/playbooks_filters.html)

## 28.6.6 Schleifen

Genauso nützlich wie das Testen von Bedingungen ist Ansibles Fähigkeit, einzelne Tasks zu wiederholen. Im Developer-Jargon nennt man das *Schleife* oder – wenn es etwas wissenschaftlicher klingen soll – *Iteration*. Iteriert werden kann u. a. über Listen oder über Maps. Sie können auch Tasks so lange wiederholen lassen, bis eine gewisse Bedingung erfüllt ist. Für diese genannten Konzepte geben wir nun Beispiele; eine vollständige Übersicht der Möglichkeiten finden Sie unter [http://docs.ansible.com/ansible/latest/user\\_guide/playbooks\\_loops.html](http://docs.ansible.com/ansible/latest/user_guide/playbooks_loops.html).

Wir schicken an dieser Stelle gleich vorweg, dass es in Ansible zwei Schleifenvarianten gibt:

- ▶ Seit jeher vorhanden:

```
with_<LOOKUP>
```

- Neuere Syntax ab Ansible 2.5:

```
loop
```

Ansible empfiehlt für die meisten Use Cases die neuere `loop`-Variante. Es gibt aber zurzeit keine Planungen, die klassische Variante nicht mehr zu unterstützen, und da der Verfasser dieser Zeilen aus praktischer Sicht keine echten Vorteile der neuen Syntax sieht, beschränken wir uns hier auf die alte Form. Machen Sie sich im Zweifelsfall in der offiziellen Dokumentation unter oben genannter URL selbst ein Bild.

### Iteration über eine Liste mit »with\_items« oder »with\_list«

Mit dem Schleifentyp `with_items` können Sie z.B. wie folgt über eine Liste iterieren:

```

- hosts: localhost
 vars:
 tiere:
 - Hund
 - Katze
 - Maus

 tasks:
 - name: Schleife über Liste
 debug: msg="Hallo {{ item }}!"
 with_items: "{{tiere}}"
```

#### Listing 28.43 »list-loop.yml«

Die Variable `item` ist gewissermaßen die »magische« Laufvariable. Sie enthält in jedem Schleifendurchlauf den aktuellen Listenwert. So sieht das Ganze aus:

```
$ ansible-playbook list-loop.yml
[...]
TASK [Schleife über Liste] *****
ok: [localhost] => (item=Hund) => {
 "msg": "Hallo Hund!"
}
ok: [localhost] => (item=Katze) => {
 "msg": "Hallo Katze!"
}
ok: [localhost] => (item=Maus) => {
 "msg": "Hallo Maus!"
}
[...]
```

### Iteration über eine Map mit »with\_dict«

Die Iteration über eine Map mit `with_dict` könnte z.B. wie folgt aussehen:

```

- hosts: localhost
 vars:
 tiere:
 Hund: Wuff
 Katze: Miau
 Maus: Fiep

 tasks:
 - name: Schleife über Map
 debug: msg="{{ item.key }} macht {{ item.value }}."
 with_dict: "{{tiere}}"
```

#### Listing 28.44 »map-loop.yml«

Auch hier gibt es wieder die magische `item`-Variable; dieses Mal besteht sie jeweils aus einem Schlüssel-Wert-Paar. Mit den Selektoren `key` bzw. `value` können Sie dann auf den gewünschten Teil zugreifen.

### Iteration über eine generierte Folge mit »with\_sequence«

Mit dem Schleifentyp `with_sequence` können Sie Folgen generieren und über sie iterieren. Ein Beispiel:

```

- hosts: localhost

 tasks:
 - name: Schleife über eine Folge
 debug: msg="{{item}}"
 with_sequence: start=6 end=12 stride=2 format=test%02d.txt
```

#### Listing 28.45 »sequence-loop.yml«

Alle Parameter von `with_sequence` sind optional und per Default 0 (das Format hat keinen Default und würde einfach Zahlen produzieren). Formate können in der üblichen `printf`-Syntax angegeben werden.

### Schleife über die Kombination zweier Listen mit »with\_nested«

```

- hosts: localhost
```



```

tasks:
 - name: Schleife über Kombination zweier Listen
 debug: msg="{{ item[0] }}" mag {{ item[1] }}"
 with_nested:
 - [Hund, Katze, Maus]
 - [Chips, Popcorn]

```

Listing 28.46 »nested-loop.yml«

```

$ ansible-playbook nested-loop.yml
[...]
TASK [Schleife über Kombination zweier Listen] *****
ok: [localhost] => (item=['Hund', 'Chips']) => {
 "msg": "Hund mag Chips"
}
ok: [localhost] => (item=['Hund', 'Popcorn']) => {
 "msg": "Hund mag Popcorn"
}
ok: [localhost] => (item=['Katze', 'Chips']) => {
 "msg": "Katze mag Chips"
}
[...]

```

28

### Schleife über zwei parallele Listen mit »with\_together«

```

- hosts: localhost

tasks:
 - name: Schleife über zwei parallele Listen
 debug: msg="{{ item[0] }}" ist die Nummer {{ item[1] }}"
 with_together:
 - [Hund, Katze, Maus]
 - [eins, zwei, drei]

```

Listing 28.47 »together-loop.yml«

```

$ ansible-playbook nested-loop.yml
[...]
TASK [Schleife über zwei parallele Listen] *****
ok: [localhost] => (item=['Hund', 'eins']) => {
 "msg": "Hund ist die Nummer eins"
}
ok: [localhost] => (item=['Katze', 'zwei']) => {
 "msg": "Katze ist die Nummer zwei"
}

```

```
}
ok: [localhost] => (item=['Maus', 'drei']) => {
 "msg": "Maus ist die Nummer drei"
}
[...]
```

### Tasks wiederholen mit »until«

Nachstehend sehen Sie ein Beispiel für den Schleifentyp until:

```

- hosts: localhost

 tasks:
 - name: Würfeln
 shell: "echo $(($RANDOM % 6 + 1))"
 register: wurf
 args:
 executable: /bin/bash
 until: "'6' in wurf.stdout"
 retries: 10
 delay: 1

 - debug: msg="Eine 6 wurde gewürfelt!"
```

**Listing 28.48** »until-loop.yml«: Task wiederholen, bis ein Ereignis eintritt

Mit diesem Schleifentyp können Sie einen Task so lange wiederholen lassen, bis eine bestimmte Bedingung eintritt. In Ansible sind aber keine Endlosschleifen möglich (also ein ewiges Warten auf eine Bedingung, die niemals eintritt). Dies wird durch den `retries`-Parameter sichergestellt, mit dem Sie eine Maximalanzahl von Versuchen festlegen müssen. Die Zeitspanne zwischen zwei Versuchen (in Sekunden) legen Sie mit `delay` fest. Der Defaultwert für `retries` ist 3 und der für `delay` ist 5.

Wenn nach `retries` Versuchen kein Erfolg eintritt, endet der Task mit einem Fehler. In diesem Fall hatte der Aufrufer aber Glück:

```
$ ansible-playbook until-loop.yml
[...]
```

```
TASK [Würfel] *****
FAILED - RETRYING: Würfeln (10 retries left).
FAILED - RETRYING: Würfeln (9 retries left).
FAILED - RETRYING: Würfeln (8 retries left).
FAILED - RETRYING: Würfeln (7 retries left).
FAILED - RETRYING: Würfeln (6 retries left).
changed: [localhost]
```

```
TASK [debug] *****
ok: [localhost] => {
 "msg": "Eine 6 wurde gewürfelt!"
}
[...]
```

### 28.6.7 Fehlerbehandlung mit »failed\_when« und »ignore\_errors«

Sollten Sie bei einem Task einmal selbst bestimmen wollen, was ein Fehler ist, so steht Ihnen die `failed_when`-Direktive zur Verfügung. Diese wird meist bei `command`- oder `shell`-Tasks zur Anwendung kommen, wenn die auszuführenden Kommandos eine andere Vorstellung von »fehlerhaft« haben als Sie selbst.

Beispielsweise ist es ja beim `find`-Kommando technisch gesehen kein Fehler, wenn nichts gefunden wird. Dies ändern wir (aus der Sicht von Ansible) in diesem Beispiel:

```

- hosts: all

 tasks:
 - name: Mit find nichts finden soll Fehler sein
 command: find /etc -name blabla.conf
 changed_when: false
 register: find_cmd
 failed_when: find_cmd.stdout == '' or find_cmd.rc != 0

 - debug: msg={{ find_cmd.stdout }}
```

#### Listing 28.49 »failed-when.yml«

Sofern nun nicht tatsächlich in irgendeinem `/etc`-Ordner eine Datei namens `blabla.conf` fliegt, wird das Play nach dem ersten Task enden. Ansible kennt noch einen weiteren Task-Parameter namens `ignore_errors`, der jedoch selten zur Anwendung kommen sollte. Wenn Sie diesen Parameter auf `yes/true` setzen, wird Ansible jegliche Fehler an dieser Stelle übergehen; die Gefahr dabei ist also, dass das Play weiterläuft, obwohl es vielleicht ein wirklich ernstes Problem gab. Hier sehen Sie ein völlig simples Beispiel:

```

- hosts: localhost

 tasks:
 - name: Dieser Task würde normalerweise fehlschlagen...
 command: /bin/false
```

```
changed_when: false
ignore_errors: true # ...deswegen aber nicht!
```

Listing 28.50 »ignore-errors.yml«

### 28.6.8 Blocks

Eines der vielen neuen Features in Ansible 2.0 war die Möglichkeit, Tasks zu logischen Blöcken zusammenzufassen. Hier folgt ein simples Beispiel, das den Nutzen sofort verdeutlichen sollte:

```

- hosts: all

 tasks:
 - block:
 - name: Nur auf Debian
 debug: msg="Nur auf Debian"

 - name: Noch was nur auf Debian
 debug: msg="Das auch nur auf Debian"

 when: ansible_os_family == "Debian"
```

Listing 28.51 »block.yml«: Zusammenfassung von Tasks zu einem Block

Nun können Sie also Parameter wie `when`, `ignore_errors`, `become_user` ... auf Block-Ebene spezifizieren, anstatt sie in jedem einzelnen Task wiederholen zu müssen. Das war sicher ein Feature, auf das die Welt damals gewartet hat. Seit Version 2.3 können Blöcke auch ein `name`-Feld haben, so wie gewöhnliche Tasks auch. Technisch hat so ein Name aber weiter keine Auswirkung, daher wird er in der Regel weggelassen.

### 28.6.9 Lookup-Plug-ins

Lookup-Plug-ins (oder kurz: Lookups) sind eine Ansible-spezifische Erweiterung von Jinja2 (siehe Abschnitt 28.6.5). Damit können Sie auf verschiedenste externe Datenquellen zugreifen, allerdings immer *ausgehend vom Control Host*. Die Ergebnisse der Lookups werden dann typischerweise in Variablen geschrieben oder direkt in Templates verarbeitet.

Sie sehen in Listing 28.52 ein erstes Beispiel: Ein `pipe`-Lookup macht die Standardausgabe eines externen Kommandos zugänglich:

```

- hosts: debian
```

```

vars:
 datetime: "{{ lookup('pipe', 'date') }}"

tasks:
 - debug: var=datetime
 - pause: seconds=2
 - debug: var=datetime

```

**Listing 28.52** »lookup\_pipe.yml«: ein »pipe«-Lookup

Der Zielhost ist in diesem Beispiel absolut egal; nichts von diesem Play passiert außerhalb des Control Hosts. Etwas unerwartet ist vielleicht die Tatsache, dass der Lookup bei jedem Zugriff auf die Variable erneut »frisch« ausgeführt wird:

```
$ ansible-playbook lookup_pipe.yml
```

```

TASK [debug] *****
ok: [debian] => {
 "datetime": "Sat 16 May 2020 07:44:38 PM UTC"
}

```

```

TASK [pause] *****
Pausing for 2 seconds
(ctrl+C then 'C' = continue early, ctrl+C then 'A' = abort)
ok: [debian]

```

```

TASK [debug] *****
ok: [debian] => {
 "datetime": "Sat 16 May 2020 07:44:40 PM UTC"
}

```

Ansible hatte in Version 2.9 etwa 60 Lookup-Plug-ins an Bord. Da aber auch die Ihnen bereits bekannten Schleifen `with_items`, `with_dict` usw. technisch als Lookup realisiert sind, dürfte die tatsächliche Anzahl so um die 50 liegen. Die komplette Übersicht finden Sie jedenfalls unter <http://docs.ansible.com/ansible/latest/plugins/lookup.html>.

In Listing 28.53 sehen Sie einige weitere Beispiele für Lookups:

```

- hosts: localhost

tasks:
 - name: File lookup
 debug:
 msg: "{{ lookup('file', '/etc/os-release') }}"

 - name: Environment lookup

```

```
debug:
 msg: "{{ lookup('env', 'HOME') }}"

- name: CSV lookup
 debug:
 msg: "{{ lookup('csvfile',
 'root file=/etc/passwd delimiter=: col=6') }}"

- name: DNS lookup
 debug:
 msg: "{{ lookup('dig', 'www.example.com') }}"
```

**Listing 28.53** »lookups.yml«: Verschiedene Lookups

### 28.6.10 Umgebungsvariablen setzen

Mit dem `environment`-Parameter können Sie Umgebungsvariablen für Zielsysteme setzen, und zwar auf der Ebene von einzelnen Tasks oder eines ganzen Plays. Eine der typischsten Verwendungen dafür ist wohl die Festlegung eines Proxys, der den Zielsystemen den Internetzugang ermöglicht (beispielsweise um Distributionspakete zu installieren). Hier sehen Sie ein fiktives Beispiel auf Task-Ebene:

```
- name: Git installieren
 package:
 name: git
 environment:
 http_proxy: http://proxy.example.org:3128
```

Auf der Play-Ebene können Sie `environment` bei Bedarf genauso spezifizieren, es gilt dann für alle Tasks im Play. Meist möchte man eine solche Einstellung aber komplett außerhalb aller Playbooks verwalten, beispielsweise in `group_vars`-Dateien. Dabei ist dann nur zu beachten, dass Sie alle benötigten Umgebungsvariablen in einer Map zusammenfassen müssen; nennen wir sie exemplarisch mal `my_env`:

```
my_env:
 http_proxy: http://proxy.example.com:3128
 https_proxy: http://proxy.example.com:3128
```

**Listing 28.54** »group\_vars/all.yml«: exemplarische Proxy-Einstellungen für alle Zielsysteme

Auf der gewünschten Ebene greifen Sie dann einfach darauf zurück:

```

- hosts: all

 environment: "{{ my_env }}"
```

```
tasks:
 [...]
```

**Listing 28.55** »some\_playbook.yml«: exemplarischer Playbook-Ausschnitt

### Suchpfad (»PATH«) erweitern

Mitunter ist es noch nötig, den Suchpfad auf einem Target Host zu erweitern, damit gewisse Programme gefunden werden. Den Ist-Zustand können Sie dabei aus dem Ansible-Fact `ansible_env` beziehen. Damit ergibt sich z. B.:

```

- hosts: all

 environment:
 PATH: "{{ ansible_env.PATH }}:/snap/bin"

 tasks:
 [...]
```

**Listing 28.56** »some\_playbook.yml«: exemplarischer Playbook-Ausschnitt

#### Anmerkung

Ein wenig Vorsicht ist hier geboten, weil das Environment ja einerseits vom Zieluser abhängt, andererseits gegebenenfalls ein Faktencache befragt wird. Sollten Sie also mal spontan z. B. einen Task mit `become_user` modifizieren, kann das merkwürdige Effekte nach sich ziehen!



## 28.7 Module und Collections verwenden

Die Module sind gewissermaßen die Arbeitstiere von Ansible. Mit ihnen lassen sich klar umrissene Aufgaben (*Tasks*) zuverlässig erledigen. Das ist ja bekanntlich auch Bestandteil der klassischen UNIX-Philosophie: Werkzeuge, mit denen man jeweils eine Sache ordentlich und richtig beherrschen kann, sind das A und O. Natürlich gibt es mitunter auch Module, die »zu viel« können, aber Ausnahmen bestätigen die Regel. Wir möchten Ihnen in diesem Abschnitt einen ersten Überblick darüber geben, welche Module Sie *typischerweise* oft brauchen werden. Viele davon haben Sie in den vorigen Abschnitten schon gesehen. Zu Beginn müssen wir aber – um eine Verständnisgrundlage zu schaffen – einen Blick auf die in Ansible relativ neuen *Collections* werfen, die u. a. zur Verteilung von Modulen genutzt werden.

### 28.7.1 Collections

Collections in Ansible sind ein Distributionsformat, mit dem Zusatzkomponenten wie Playbooks, Module, Rollen und Plug-ins auf standardisierte Art und Weise verpackt und ange-

boten werden können. Sie sind noch ein relativ neuer Bestandteil von Ansible, der mit Version 2.10 vollständig umgesetzt wurde. Mit ein Grund für die Einführung von Collections war, dass das alte Entwicklungskonzept von Ansible (»Alles was man braucht, ist im Lieferumfang enthalten«, auch: »Batteries Included«) irgendwann nicht mehr tragfähig war. Es wurde immer schwieriger, mit den zu dieser Zeit weit über 3000 Modulen, die zum großen Teil von der Community gepflegt werden, ein vernünftiges Release hinzubekommen.

Also trennte man den Kern von Ansible (*ansible-core*) und die von der Community beigesteuerten Module und Plug-ins in unterschiedliche Entwicklungspfade auf. In Abschnitt 28.1.2 hatte ich das bereits etwas genauer aufgeschlüsselt.

### Eine Minimalumgebung mit »ansible-core«

Wenn Sie Interesse haben, sehen wir uns das einmal in der Praxis an. Mit einer Virtualenv-Umgebung können Sie das risikofrei tun, ohne unserer produktiven Installation in die Quere zu kommen.

Richten wir zunächst eine Virtualenv-Umgebung ein (siehe dazu auch Abschnitt 28.1.4, Unterabschnitt »Installation via PIP (+Virtualenv)«). Auch an dieser Stelle sei noch einmal betont, dass Sie dafür keine Root-Rechte benötigen:

```
$ python3 -m venv ~/venv/ansible-core
$ source ~/venv/ansible-core/bin/activate
```

Nun können Sie das aktuelle stabile *ansible-core*-Paket installieren:

```
$ pip3 install ansible-core
```

Das geht relativ schnell. Ein Grund ist sicher, dass wir nun statt mehreren Tausend Modulen nur noch ein paar Dutzend wichtige Basismodule mit an Bord haben:

```
$ ansible-doc -l | wc -l
69
```

Wenn Sie `| wc -l` weglassen, können Sie sich die Liste auch gern anschauen. Es sind viele alte Bekannte dabei. Möglicherweise sehen Sie aber auch auf den zweiten Blick, dass viele Ihrer Lieblingsmodule fehlen. Ein beliebiges Beispiel für ein Modul, das in einer »normalen« Ansible-Installation enthalten ist, jedoch hier nicht, ist *ini\_file*:

```
$ ansible-doc ini_file
[WARNING]: module ini_file not found in:
[...]
```

Dieses Modul findet sich mittlerweile in der Collection *community.general*, was Sie der Seite [http://docs.ansible.com/ansible/latest/collections/community/general/ini\\_file\\_module.html](http://docs.ansible.com/ansible/latest/collections/community/general/ini_file_module.html) entnehmen können, die Sie bei einer Recherche sofort finden würden.



## Collections installieren

Mit dem Kommando `ansible-galaxy` können Sie u. a. Collections direkt aus dem Internet installieren. Der einfachste Aufruf dafür wäre:

```
$ ansible-galaxy collection install <COLLECTION_NAME>
```

Allerdings möchten wir das bei unserem kleinen Test nicht in der Default-Location `~/.ansible/collections` tun, um die produktive Installation nicht zu beeinflussen. Also legen wir ein neues Verzeichnis an, z. B.:

```
$ mkdir -p ~/coredemo/collections
```

Dieses Verzeichnis kann nun als Ziel der Installation gesetzt werden:

```
$ ansible-galaxy collection install -p ~/coredemo/collections community.general
```

Damit Ansible diese Stelle auch bei der Suche nach Collections berücksichtigt, können wir (in unserem Fall temporär) mit der Umgebungsvariablen `ANSIBLE_COLLECTIONS_PATHS` arbeiten:

```
$ export ANSIBLE_COLLECTIONS_PATHS=~/coredemo/collections
```

Das Modul `ini_file` (und sehr viele andere) stehen nun wieder wie gewohnt zur Verfügung:

```
$ ansible-doc ini_file
```

```
$ ansible-doc -l | wc -l
602
```

Einen Überblick über alle installierten Collections erhalten Sie übrigens mit:

```
$ ansible-galaxy collection list
```

Wenn Sie diesen kleinen Ausflug mitgemacht haben, können Sie nun die Virtualenv-Umgebung mit `deactivate` wieder verlassen.

## Den Collection-Suchpfad per Ansible-Konfiguration erweitern

Im vorigen Abschnitt haben Sie die Umgebungsvariable `ANSIBLE_COLLECTIONS_PATHS` kennengelernt, mit der Sie den Collection-Suchpfad setzen können. Alternativ dazu gibt es natürlich auch einen entsprechenden Konfigurationsparameter; er heißt erwartungsgemäß `collections_paths`. Leider gibt es aber keine einfache Methode, den Suchpfad nur zu *erweitern*. Sie müssen dazu den aktuellen Stand herausfinden und Ihren Pfad bzw. Ihre Pfade ergänzen. Der Ansible-Default ist typischerweise `~/.ansible/collections:/usr/share/ansible/collections`; verifizieren Sie das aber noch einmal, beispielsweise mit:

```
$ ansible-config dump | grep COLLECTIONS_PATHS
```

Möchten Sie nun in Ihrem Projekt beispielsweise den Suchpfad um den Ordner `~/coredemo/collections` erweitern, dann könnte die Konfiguration so aussehen:

```
[defaults]
collections_paths = ~/.ansible/collections:/usr/share/ansible/collections:↵
~/coredemo/collections
```

**Listing 28.57** »ansible.cfg«: Collection-Suchpfad erweitern



### Hinweis

In diesem Dateiformat gibt es leider keine korrekte Möglichkeit, überlange Zeilen umzubrechchen. Das Symbol ↵ soll deswegen anzeigen, dass die folgende Zeile in Wirklichkeit eine Fortsetzung der aktuellen Zeile ist.

### Der FQCN (Fully Qualified Collection Name)

Vor allem, wenn Sie schon etwas länger mit Ansible arbeiten, ist Ihnen vielleicht aufgefallen, dass in der Dokumentation mittlerweile stets mit vollqualifizierten Modulnamen gearbeitet wird, also beispielsweise mit

```
- community.general.ini_file:
 path: /tmp/test.ini
 [...]
```

anstatt (wie früher) mit:

```
- ini_file:
 path: /tmp/test.ini
 [...]
```

Ansible setzt damit ein typisches Namespace-Konzept um, das in erster Linie Namenskollisionen vermeidet. Es könnte ja theoretisch noch eine andere Collection geben, die ebenfalls ein Modul namens `ini_file` enthält. Aktuell verwendet bei »Standardmodulen« jedoch kaum jemand die lange Form, obwohl die offizielle Dokumentation das sehr nahelegt. Der einfache Grund dafür ist, dass die Kurzform nach wie vor funktioniert. Dafür mussten die Ansible-Entwickler natürlich sorgen, ansonsten hätten sehr viele Leute Tausende von Tasks in Hunderten von Playbooks und Rollen ändern müssen, was sehr viele Leute sehr ärgerlich gemacht hätte. Deswegen wurde entschieden, dass alle Module, Plug-ins und sonstige Objekte, die bis zum Stichtag der Umstellung im alten Ansible enthalten waren, auch in Zukunft mit der kurzen Form ansprechbar bleiben. Aber auch wenn Sie Collections einsetzen, die nicht in dieser Ausnahmeregelung enthalten sind, können Sie auf Playbook- oder Rollenebene mit dem Schlüsselwort `collections` alle Namespaces auflisten, in denen Ansible partiell qualifizierte Objekte suchen soll. Das tun Sie aber bitte nur dann, wenn Sie

- ▶ überhaupt keine Lust auf vollqualifizierte (= lange) Namen haben und
- ▶ das Risiko einer Namenskollision als sehr überschaubar einschätzen.

Konkret sähe das so aus:

### Erleichterte Verwendung von Collections in Playbooks

```
- hosts: all

collections:
 - namespace1.eine_collection
 - namespace2.andere_collection
```

**Listing 28.58** »playbook.yml«: Objekte aus Collections mit kurzem Namen ansprechbar machen

### Zwischenfazit

Solange Sie ein Community-Package von Ansible verwenden und mit dem Lieferumfang zufrieden sind, brauchen Sie sich keine allzu großen Gedanken um Collections zu machen. Aber auch das Nachinstallieren von Collections wäre ja kein Problem, wie wir gesehen haben.

Sollten Sie aber irgendwann eigene Ansible-Komponenten wie Module oder Plug-ins entwickeln *und* diese in einem standardisierten Format verteilen wollen, wird das Thema durchaus um einiges interessanter. Damit kommen wir also endlich zum eigentlichen Anliegen dieses Abschnitts: den Modulen.

## 28.7.2 Module

Ein Ansible-Community-Package enthält (je nach Version) ca. 4000 bis 6000 Module. Eine Übersicht über alle lokal verfügbaren Module erhalten Sie mit:

```
$ ansible-doc -l
```

Sie können eine Modul-Dokumentation auch offline lesen mit dem Befehl:

```
$ ansible-doc <MODULNAME>
```

Online gibt es aufgrund der Neustrukturierung mittels Collections keine einfache Übersichtsseite mehr. Einen recht umfassenden Index finden Sie unter <http://docs.ansible.com/ansible/latest/collections/>. Solange er noch verfügbar ist, können Sie auch den Modul-Index der Version 2.9 verwenden, der recht übersichtlich nach Kategorien sortiert ist: [https://docs.ansible.com/ansible/2.9/modules/modules\\_by\\_category.html](https://docs.ansible.com/ansible/2.9/modules/modules_by_category.html)

Generell sollten Sie bei Online-Dokumentationen aber immer darauf achten, dass die Dokumentation, die Sie gerade lesen, auch zu Ihrer eingesetzten Ansible-Version passt!

Wir möchten Ihnen in den folgenden Abschnitten eine übersichtliche Auswahl von Modulen vorstellen und geben zu jedem Modul ein oder zwei Aufrufbeispiele: ein recht kurzes und gegebenenfalls eines, das weitergehende Aufrufmöglichkeiten demonstriert. Einige Module werden wir auch nur erwähnen. Ohnehin gilt: Wenn ein Modul für Sie interessant oder wich-

tig ist, dann ziehen Sie im Zweifel stets zusätzlich die offizielle Dokumentation zurate. Wir sparen uns aus Platzgründen die Angabe der jeweiligen URL zur offiziellen Dokumentationsseite bei jedem einzelnen Modul. Mit einer Suchmaschine Ihrer Wahl und dem Suchtext »Ansible Module <MODULNAME>« sind Sie ohnehin schneller.

Sollte ein Modul nicht aus dem Ansible-Core bzw. `ansible.builtin` stammen, werden wir es mit seinem FQCN angeben. Falls es auch nicht im Lieferumfang des Ansible-Community-Packages ist, werden wir darauf noch einmal gesondert hinweisen.

### 28.7.3 Module zur Kommandoausführung

Zur Kommandoausführung stehen die folgenden Module zur Verfügung:

#### **command – Kommando ausführen**

- name: Dateien suchen und Ergebnis in Variable sichern  
command: `find /etc -name *.conf`  
register: `find_cmd`
  
- name: Ergebnis einer Sortierung in /tmp ablegen  
command:  
  chdir: `/tmp`  
  cmd: `sort /etc/passwd -o sorted.txt`  
  creates: `sorted.txt`

Das zweite Beispiel zeigt den Wechsel des Arbeitsverzeichnisses mit `chdir` und einen »Idempotenzhinweis« mit `creates`.

#### **shell – Kommando über eine Shell ausführen**

- name: Programme mit "a" am Anfang zählen  
shell: `"ls -l /usr/bin/a* | wc -l"`
  
- name: Brace-Expansion (benötigt eine Bash)  
shell:  
  cmd: `echo test{1,2,3}`  
  executable: `/bin/bash`

Dieses Modul ähnelt dem `command`-Modul, nur wird der Befehl hier per Default via `/bin/sh` ausgeführt. Wenn Sie Shell-Mechanismen benötigen, die von einer klassischen Bourne-Shell nicht unterstützt werden, können Sie den gewünschten Shell-Interpreter auch angeben (siehe das zweite Beispiel).

#### **raw – Kommando ohne Python-Subsystem ausführen**

- name: Python 3 auf einem Debian-Target-Host installieren  
raw: `apt-get -y install python3`

Am häufigsten wird dieses Modul wahrscheinlich eingesetzt, um auf einem Zielsystem ohne Python überhaupt erst mal ein Python-Paket zu installieren. Aus technischer Sicht ist es mit dem `shell`-Modul vergleichbar; der `executable`-Parameter steht hier ebenfalls zur Verfügung.

### **script – lokales Skript übertragen und ausführen**

```
- name: Skript übertragen und ausführen
 script: /usr/local/bin/myscript.sh --opt1 1234
```

Der Name dieses Moduls ist etwas irreführend, da Sie jede Art von Programm verwenden können. Natürlich werden es in der Praxis meist Shell-, Perl- oder Python-Skripte sein. Interessant ist noch, dass (genau wie beim `raw`-Modul) kein Python auf der Zielmaschine vorausgesetzt wird. Ansonsten sind die Parametrisierungsmöglichkeiten vergleichbar mit denen des `shell`-Moduls.

28

## **28.7.4 Module zur Paketverwaltung**

Die folgenden Module dienen zur Paketverwaltung.

### **apt – Paketmanagement auf Debian/Ubuntu-Systemen**

```
- name: Das Pendant zu apt-get update
 apt:
 update_cache: yes
 cache_valid_time: 3600

- name: Paket "tmux" installieren
 apt:
 name: tmux
```

Zur Verwaltung von Repositories und Schlüsseln stehen die Module `apt_repository` und `apt_key` zur Verfügung.

### **dnf, yum – Paketmanagement auf Rocky-/CentOS-/Red-Hat-/Fedora-Systemen**

```
- name: Paket "tree" installieren
 dnf:
 name: tree

- name: Paket "tmux" installieren
 yum:
 name: tmux
```

Zur Verwaltung von Repositories steht das Modul `yum_repository` zur Verfügung.

### **community.general.zypper – Paketmanagement auf SUSE-Systemen**

- name: Paket "tmux" installieren  
zypper:  
  name: tmux

Zur Verwaltung von Repositories steht das Modul `zypper_repository` zur Verfügung.

### **package – generisches Paketmanagement**

- name: Paket "tmux" installieren  
package:  
  name: tmux

Das `package`-Modul ist der kleinste gemeinsame Nenner des Paketmanagements. Unter der Haube wird einfach das eigentliche Modul (`apt`, `dnf` etc.) aufgerufen.

### **package\_facts – Paketinformationen als Fakten darstellen**

- name: Fakten über installierte Pakete einsammeln  
package\_facts:
- debug: msg="Apache ist installiert"  
  when: ansible\_facts['packages']['apache2'] is defined
- debug: var=ansible\_facts['packages']['apache2']

Das Modul `package_facts` integriert Informationen über installierte Distributionspakete in die Fakten. Die Datenstruktur pro Paket sieht für obigen Fall z.B. so aus:

```
"ansible_facts['packages']['apache2']": [
 {
 "arch": "amd64",
 "category": "httpd",
 "name": "apache2",
 "origin": "Debian",
 "source": "apt",
 "version": "2.4.38-3+deb10u3"
 }
]
```

### **Einige weitere (nicht näher beschriebene) Module**

- ▶ `pip` – Python-Module verwalten
- ▶ `community.general.cpanm` – Perl-Module verwalten
- ▶ `community.general.pear` – PHP-Pakete verwalten

### 28.7.5 Module zur Verwaltung von Dateien und Dateiinhalten

Mit den folgenden Modulen können Sie sowohl Dateien als auch Dateiinhalte verwalten.

#### copy – Dateien kopieren bzw. hochladen

- name: Beispieldatei "test1.txt" zum Zielhost kopieren
 

```
copy:
 src: test1.txt
 dest: /tmp
```
- name: Dabei auch Zugriffs- und Eigentumsrechte verwalten
 

```
copy:
 src: test1.txt
 dest: /tmp
 owner: user1
 group: users
 mode: 0600
```

Wenn die Quelldatei durch einen relativen Pfad spezifiziert wird, sucht Ansible in einem Ordner *files/* parallel zum Playbook und danach direkt parallel zum Playbook. Bei Verwendung in einer Rolle wird zunächst im Rollenordner *files/* gesucht. Sollte sich die zu kopierende Datei bereits auf dem Zielsystem befinden, müssen Sie dies mit dem Parameter `remote_src: yes` mitteilen. Der `mode` kann numerisch oder symbolisch (z. B. `'u+rw'`) angegeben werden. Außerdem steht das Schlüsselwort `preserve` zur Verfügung, mit dem die Zieldatei dieselben Rechte wie die Quelldatei bekommt.

#### Achtung

Es gibt zwei korrekte Möglichkeiten, einen numerischen Mode anzugeben:

- ▶ mithilfe einer Zahl mit führender 0 (wie im obigen Beispiel), sodass der YAML-Parser von Ansible sofort eine Oktalzahl erkennt
- ▶ mit einem *quotierten* String wie `'644'` oder `'1777'`, sodass Ansible intern selbst konvertieren kann

Wenn Sie eine Zahl anders spezifizieren, wird dies nicht richtig funktionieren!

#### template – Dateien mit Jinja verarbeiten und hochladen

- name: Template füllen und hochladen
 

```
template:
 src: index.html.j2
 dest: /srv/www/htdocs/index.html
```



Dieses Modul ist in der Verwendung nahezu identisch mit dem `copy`-Modul, nur dass der Input zusätzlich mit der Jinja-Engine verarbeitet wird. (Falls Ihnen der Begriff »Jinja« nichts sagt, sollten Sie in jedem Fall Abschnitt 28.6.5 lesen.) Beachten Sie bitte, dass bei relativen Pfadangaben hier zusätzlich im Ordner `templates/` nach der Quelldatei gesucht wird (und *nicht* in `files/!`).

### file – Dateien und Dateiattribute verwalten

```
- name: Zugriffs- und Eigentumsrechte setzen
 file:
 path: /etc/ssl/private
 owner: root
 group: ssl-cert
 mode: 0710
```

Mit `file` können Sie auch Dateien und Verzeichnisse anlegen oder löschen, Soft- und Hardlinks verwalten und einiges mehr.

Gerade das Anlegen von Links geht mit diesem Modul aber nicht immer völlig leicht von der Hand, deswegen dazu noch einige Bemerkungen. Angenommen, Sie möchten mit Ansible erreichen, was Sie in der Shell wie folgt lösen:

```
$ ln -s <ALTER_NAME> <NEUER_NAME>
```

So merken sich viele Linux-User auch die Reihenfolge der Parameter beim `ln`-Kommando: Erst kommt der Pfad zur alten (also meist schon existierenden) Datei, dann kommt der Linkname – also ein neuer Name für diese Datei. Der entsprechende Ansible-Task sieht dann so aus:

```
- name: Symbolischen Link setzen
 file:
 src: ALTER_NAME
 dest: NEUER_NAME
 state: link
```

Einen Hardlink bekämen Sie mit `state: hard`. Die Syntax des `file`-Moduls kommt Ihnen also hier entgegen, wenn Sie sich die Funktion des `ln`-Kommandos sowieso mit »Quelle« und »Ziel« merken. Problematisch wird es nun noch in zwei Situationen:

- ▶ Die alte Datei bzw. die Quelle existiert nicht (das wäre ja für das `ln`-Kommando bei Symbolinks kein Problem).
- ▶ Der neue Name, also der Link-Pfad, existiert bereits, ist aber kein Symlink (sondern ein Verzeichnis, eine normale Datei oder was auch immer).

Diese Fälle wären für Ansible jeweils ein Fehler, den Sie aber mit dem zusätzlichen Modul-Parameter `force: yes` vermeiden können.



**stat – Informationen über Dateien gewinnen**

- name: Dateieigenschaften von /etc/ssl bestimmen
 

```
stat:
 path: /etc/ssl
 register: p
```
- debug:
 

```
msg: /etc/ssl ist ein existierendes Verzeichnis
when: p.stat.exists and p.stat.isdir
```

Ein Tipp: Schauen Sie sich einmal die komplette Struktur der registrierten Variablen an (debug: var=p). Da ist noch einiges mehr an Informationen enthalten.

28

**lineinfile – Zeilen in Textdateien verwalten**

- name: Zeile ändern
 

```
lineinfile:
 path: /etc/selinux/config
 regexp: '^SELINUX='
 line: 'SELINUX=disabled'
```
- name: Zeile an festgelegter Stelle einfügen
 

```
lineinfile:
 path: /etc/ssh/sshd_config
 insertafter: '^#ListenAddress'
 regexp: '^ListenAddress'
 line: 'ListenAddress 192.168.150.20'
```

Für einen erfolgreichen Einsatz dieses Moduls (sowie der Module `blockinfile` und `replace`) müssen Sie einigermaßen sattelfest im Umgang mit regulären Ausdrücken sein. Für die absoluten Feinheiten des konkreten »Dialekts« gehen Sie bitte auf <http://docs.python.org/3/library/re.html>.

**blockinfile – Textpassagen in Dateien verwalten**

- name: Konfiguration für virtuelles Interface eintragen
 

```
blockinfile:
 path: /etc/network/interfaces
 block: |
 iface ens33:0 inet static
 address 10.0.0.1/24
 netmask 255.255.255.0
```

Das Modul verwaltet Textpassagen, die von konfigurierbaren Markierungszeilen umgeben sind. Zur Positionierung stehen mit `lineinfile` vergleichbare Möglichkeiten zur Verfügung.

### replace – Suchen und Ersetzen in Textdateien

```
- name: PasswordAuthentication ggf. einkommentieren und auf "no" setzen
 replace:
 path: /etc/ssh/sshd_config
 regexp: '^#?(PasswordAuthentication).*'
 replace: '\1 no'
```

Das Modul ersetzt *alle* Vorkommen von `regexp` in der Datei durch den `replace`-Ausdruck. Falls `replace` fehlt, werden die Vorkommen gelöscht.

### unarchive – Archive hochladen und auspacken

```
- name: Archiv-Datei in /srv/www/webapps auspacken
 unarchive:
 src: 'roundcubemail-1.3.4-complete.tar.gz'
 dest: /srv/www/webapps
 owner: www-data
 group: www-data
 creates: '/srv/www/webapps/roundcubemail-1.3.4'
```

Dieses Modul kann mit den Archivformaten `.zip`, `.tar`, `.tar.gz`, `.tar.bz2` und `.tar.xz` umgehen. Wenn die Quelle als relativer Pfad spezifiziert wird, ist die Suchstrategie dieselbe wie beim `copy`-Modul. Die Zielverzeichnisse müssen existieren. Genau wie beim `copy`-Modul steht auch hier der Parameter `remote_src` zur Verfügung, den Sie auf `yes` setzen müssen, falls sich die Archivdatei bereits auf dem Zielsystem befindet.

### Einige weitere (nicht näher beschriebene) Module

- ▶ `ansible.posix.acl` – File-ACLs verwalten
- ▶ `assemble` – Dateifragmente zusammensetzen
- ▶ `fetch` – wie `copy`, nur anders herum
- ▶ `find` – das `find`-Kommando als Modul
- ▶ `community.general.ini_file` – Inhalte von INI-Dateien verwalten
- ▶ `community.general.pamd` – PAM-Konfiguration verwalten
- ▶ `ansible.posix.synchronize` – das `rsync`-Kommando als Modul
- ▶ `tempfile` – temporäre Dateien oder Verzeichnisse erzeugen
- ▶ `community.general.xml` – Inhalte von XML-Dateien verwalten

## 28.7.6 Module für weitere typische Verwaltungsaufgaben

Für weitere typische Verwaltungsaufgaben stehen Ihnen u. a. folgende Module zur Verfügung:

**service – Dienste starten, stoppen, neu starten, ...**

- name: Apache starten und in Autostart integrieren
 

```
service:
 name: httpd
 state: started
 enabled: yes
```

**service\_facts – Service-Informationen als Fakten darstellen**

- name: Fakten über Dienste einsammeln
 

```
service_facts:
```
- debug: msg="Apache läuft"
 

```
when: ansible_facts['services']['apache2']['state'] == 'running'
```
- debug: var=ansible\_facts['services']['apache2']

Das Modul `service_facts` integriert Informationen über installierte Distributionspakete in die Fakten. Die Datenstruktur pro Dienst sieht für den obigen Fall z.B. so aus:

```
"ansible_facts['services']['apache2']": {
 "name": "apache2",
 "source": "sysv",
 "state": "running"
}
```

**cron – Cronjobs verwalten**

- name: Cronjob unter `/etc/cron.d/` einrichten
 

```
cron:
 cron_file: certbot
 user: root
 name: Run "certbot renew" once per week
 special_time: weekly
 job: certbot renew --post-hook "systemctl restart postfix apache2"
```

Natürlich stehen zur Zeitspezifikation auch Parameter wie `minute`, `hour`, `day` ... zur Verfügung.

**hostname – den Hostnamen ändern**

- name: Hostnamen auf "web01" setzen
 

```
hostname:
 name: web01
```

### user, group – Benutzer- und Gruppenverwaltung

- name: Gruppe "vmail" anlegen

```
group:
 name: vmail
 gid: 5000
```
- name: User "vmail" anlegen

```
user:
 name: vmail
 uid: 5000
 group: vmail
 home: /srv/imap
 shell: /bin/false
```

### ansible.posix.firewalld – Firewalld-basierte Firewalls verwalten

- name: HTTP-Port 80/tcp öffnen

```
firewalld:
 service: http
 state: enabled
 permanent: yes
 immediate: yes
```

Dieses Modul benötigt auf den Zielsystemen entsprechende Python-Bindings zum Firewalld-Management. In seltenen Fällen müssten Sie also vorher das Paket *python3-firewall* nachinstallieren. Ansonsten muss man an dieser Stelle sagen: Ohne fundiertes Wissen zu Firewalld wird man auch mit diesem Modul nicht besonders weit kommen!

### reboot – Maschinen rebooten

- name: Maschine neu starten

```
reboot:
```

### Einige weitere (nicht näher beschriebene) Module

- ▶ `ansible.posix.authorized_key` – SSH-Keys verwalten
- ▶ `community.general.filesystem` – Dateisysteme erzeugen
- ▶ `getent` – Wrapper um das Linux-getent-Kommando
- ▶ `community.general.locale_gen` – Locales verwalten
- ▶ `community.general.lvg, .lvol` – LVM-Devices und -Volumegruppen bzw. LogicalVolumes verwalten
- ▶ `ansible.posix.mount` – Mounts verwalten

### 28.7.7 Spezialmodule (Kontrollflusssteuerung etc.)

In diesem Abschnitt finden Sie eine Auswahl von Modulen, mit denen Sie sehr spezielle (meist interne) Aktionen auslösen können.

#### meta – Verschiedene Ansible-Aktionen auslösen

```
- name: Wir beenden das jetzt
 meta: end_play
```

Die Menge der verfügbaren Meta-Aktionen ist im Laufe der Jahre immer weiter angewachsen. Ansible  $\geq$  2.8 kennt acht Aktionen:

- ▶ **clear\_facts**  
Fakten und Fakten-Cache löschen
- ▶ **clear\_host\_errors**  
Sehr missverständlich – nimmt lediglich Unreachable Hosts wieder mit an Bord, und das auch nur, wenn überhaupt noch ein Host übrig geblieben ist.
- ▶ **end\_host**  
Beendet das Play für den aktuellen Host (ohne Fehler).
- ▶ **end\_play**  
Beendet das Play für alle Hosts (ohne Fehler). Verknüpfen Sie `end_play` nicht mit *host-spezifischen* Bedingungen (z.B. `when: inventory_hostname == "server27"`). Das Verhalten ist dann ziemlich unvorhersagbar, und in keinem Fall erreichen Sie, was Sie beabsichtigt haben. Nutzen Sie zu diesem Zweck `end_host`!
- ▶ **flush\_handlers**  
»Vorzeitiger« Aufruf von Handlern; siehe Abschnitt 28.5.8.
- ▶ **noop**  
Tut nichts (das muss auch mal sein).
- ▶ **refresh\_inventory**  
Löst einen Reload des Inventorys aus (ergibt nur Sinn bei dynamischen Inventorys).
- ▶ **reset\_connection**  
Wirft die persistenten SSH-Verbindungen weg, sodass eventuelle Folge-Tasks dann neue Verbindungen aufbauen würden.

#### debug – beim Playbook-Lauf Ausgaben erzeugen

```
- debug: msg="Hallo"

- debug: var=ansible_default_ipv4
```

Auf die Benennung der Tasks mit `name` wird sehr oft verzichtet, schon allein, weil ein `debug`-Task oft nicht dauerhaft im Play verbleiben soll. Der Parameter `var` ist dann praktisch, wenn Sie möglichst einfach den Inhalt einer Variablen sichtbar machen möchten.

### fail – Fehlschlag auslösen

- set\_fact:  
hour: "{{ lookup('pipe', 'date +%H') }}"
- fail:  
msg: "Es ist heute einfach schon zu spät."  
when: hour|int > 20

Mit dem fail-Modul können Sie gezielt einen Fehlschlag auslösen und das Play damit für die jeweiligen Hosts beenden. In den alten Ansible-Tagen, bevor meta: end\_play erfunden wurde, war dies die einzige Möglichkeit, aus einem Play auszusteigen.

### pause – pausieren und optional Eingaben einlesen

- pause:  
prompt: Bitte machen Sie nun das Licht im Serverraum aus
- name: Der CPU Zeit zum Abkühlen geben  
pause:  
seconds: 30
- pause:  
prompt: Geben Sie bitte noch Ihren Namen ein  
register: this
- debug: msg="Auf Wiedersehen, {{this.user\_input}}!"

Mit dem pause-Modul können Sie die Ausführung eines Playbooks anhalten bzw. für eine gewisse Zeit pausieren.



#### Achtung

Bitte wenden Sie das Warten auf interaktive Eingaben *nicht* an, wenn Ihr Playbook eventuell in nichtinteraktiven Umgebungen wie Cron oder Ansible AWX/Tower ablaufen soll!

### wait\_for – auf gewisse Ereignisse warten

- name: Warten, bis die Datei /tmp/foo auftaucht  
wait\_for:  
path: /tmp/foo
- name: Warte max. 300 Sekunden, bis Port 8000 offen ist  
wait\_for:  
port: 8000

Mit dem `wait_for`-Modul können Sie im Play warten, bis eine gewisse Bedingung eintritt. Die maximale Wartezeit beträgt per Default 300 Sekunden (Parameter `timeout`). Neben den gezeigten Möglichkeiten können Sie auch auf das Schließen eines Ports warten, oder Sie können Dateinhalte oder Sockets mit einem regulären Ausdruck vergleichen und dabei warten, bis das Muster passt.

#### **wait\_for\_connection – warten, bis ein Target Host erreichbar ist**

```
- name: Warte max. 600 Sekunden, bis der Target Host erreichbar ist
 wait_for_connection:
 timeout: 600
```

Das Timeout von 600 Sekunden ist aber ohnehin der Default und müsste daher nicht eigens angegeben werden.

28

#### **assert – sicherstellen, dass gewisse Bedingungen erfüllt sind**

```
- name: Es muss ein Debian 9 oder 10 sein
 assert:
 that:
 - ansible_distribution == "Debian"
 - ansible_distribution_major_version is version('9', '>=')
 - ansible_distribution_major_version is version('10', '<=')
 fail_msg: "Wir unterstützen leider nur Debian in Version 9 oder 10"
```

Das `assert`-Modul stellt sicher, dass alle angegebenen Bedingungen zutreffen, ansonsten bricht die Verarbeitung mit einem Fehler ab.

#### **set\_fact – während der Laufzeit Variablen setzen**

```
- name: Zwei Variablen setzen
 set_fact:
 foo: 100
 bar: 200

- debug: msg="foo = {{foo}}, bar = {{bar}}"
```

Mit `set_fact` können Sie während der Laufzeit host-spezifische Variablen setzen. Lesen Sie dazu gegebenenfalls Abschnitt 28.6.1.

## **28.8 Nächste Schritte**

Wie bei sehr vielen Themen, die in diesem Buch besprochen werden, steht uns auch beim Thema Ansible nur ein begrenzter Seitenumfang zur Verfügung, sodass wir an dieser Stelle nicht alle Möglichkeiten dieser schönen Software ausführlich beschreiben können. Mit den

bisher besprochenen Techniken können Sie aber erfahrungsgemäß schon eine ganze Menge Probleme lösen, und Sie haben eine solide Grundlage, um auf eigene Faust weiterzukommen. Wenn Ihre Ansible-Projekte ambitionierter werden, könnten die folgenden Themen bald für Sie relevant sein:

► **Rollen**

Würden Sie mit den bisherigen Mitteln reale Server konfigurieren, so hätten Sie schnell Playbooks, die mehrere Hundert Tasks enthalten. Das wäre aber weder in puncto Übersicht noch in puncto Wartbarkeit und Wiederverwendbarkeit wünschenswert.

Um diesem Problem zu begegnen, können Sie *Rollen* entwickeln und nutzen. Diese sind in Ansible der primäre Mechanismus, um die Tasks eines Playbooks thematisch und wiederverwendbar zu organisieren.

Sofern Sie gern mit der Online-Dokumentation von Ansible arbeiten, können Sie ausgehend von der Seite [http://docs.ansible.com/ansible/latest/playbook\\_guide/playbooks\\_reuse\\_roles.html](http://docs.ansible.com/ansible/latest/playbook_guide/playbooks_reuse_roles.html) einen guten Überblick bekommen.

► **Ansible Vault**

Früher oder später benötigen Sie eine Strategie für den Umgang mit sensiblen Daten wie Passwörtern, privaten Schlüsseln o. Ä.. Wenn Sie mit Ansible solche Daten verarbeiten wollen, sollen bzw. dürfen diese aus vielerlei Gründen nicht im Klartext in irgendwelchen Projektdateien auftauchen. Beispielsweise landen sie dann ganz schnell im Versionskontrollsystem, womit die Gefahr einer »Veröffentlichung« schon deutlich höher ist.

Der Lösungsvorschlag von Ansible ist die *Ansible Vault* bzw. das Kommando `ansible-vault`. Auch hier belassen wir es mit einem Verweis auf die Online-Dokumentation: [http://docs.ansible.com/ansible/latest/vault\\_guide/](http://docs.ansible.com/ansible/latest/vault_guide/)

Wie es darüber hinaus weitergeht, hängt sehr davon ab, was Sie machen möchten. Es gibt verschiedene Webinterfaces zu Ansible, falls Sie einen bequemerem und besser verwaltbaren Zugang zu Ihrer Infrastruktur benötigen. Vielleicht arbeiten Sie auch in der Cloud; dort kann Ansible sogar die benötigte Infrastruktur provisionieren. Vielleicht ist Windows bei Ihnen ein Thema? Auch mit diesem Betriebssystem ausgestattete Maschinen können Sie mit Ansible managen.

Am Ende soll auch nicht unerwähnt bleiben, dass in unserem Verlag auch ein Buch zum Thema Ansible erschienen ist, das diesen ganzen Themen (und noch vielen weiteren) natürlich erheblich mehr Raum gibt: »Axel Miesen: Ansible. Das Praxisbuch für Administratoren und DevOps-Teams« (Rheinwerk Verlag, 2022).



# Kapitel 29

## Monitoring – wissen, was läuft

*Checkmk ist ein mächtiges Monitoring-Framework, das sich sowohl für kleine als auch für extrem komplizierte große Setups eignet. Ganz besonders toll: Ist es einmal grundlegend aufgesetzt, kann es automatisch bestimmen, welche Details auf den Systemen überwacht werden. Denn das schönste Monitoring-System nützt nichts, wenn es nur unvollständig Dienste und Komponenten im Blick behält. Dieses Kapitel führt Sie in die Installation, Denkweise und Konfiguration von Monitoring-Systemen am Beispiel von Checkmk ein.*

Vor vielen Jahren war *Nagios* noch ein Synonym für Monitoring. Doch das hat sich geändert: Aus *Nagios* heraus und um *Nagios* herum haben sich zahlreiche weitere Lösungen entwickelt. Auch *Checkmk* startete im Jahr 2008 einst als *Nagios*-Plug-in, doch seit etlichen Jahren haben sich die Wege der beiden getrennt: *Checkmk* ist kein bloßes Plug-in mehr, sondern bringt einen eigenen Monitoring-Server mit eigener Oberfläche mit, auch wenn die Konzepte unverkennbar die gleichen Ursprünge haben. In der Open-Source-Edition benutzt *Checkmk* weiterhin *Nagios* als Monitoring-Kern, die Enterprise-Fassung bringt einen optimierten eigenen *Checkmk Micro Core* mit.

### 29.1 Monitoring mit Checkmk

Ein *Checkmk*-System besteht aus zwei Komponenten:

- ▶ aus dem *Checkmk*-Server, der Ihre Netzwerke, Hosts und andere Systeme überwacht,
- ▶ und aus einem *Checkmk*-Agent, der auf jedem einzelnen zu überwachenden Host installiert ist und lokale Informationen dem Monitoring-Server zur Verfügung stellt.

Beginnen wir mit der Installation des Servers.

### 29.2 Installation der Pakete

Auch wenn Ihre Distribution vielleicht eigene *Checkmk*-Pakete mitbringt, sollten Sie sich zuerst auf der *Checkmk*-Webseite nach der jeweils neuesten Version erkundigen. Der neue *Checkmk*-Versionszweig 2.x vereinfacht die Benutzerführung gegenüber dem bisherigen

Zweig 1.x und sollte damit gerade für Einsteiger leichter zu handhaben sein. Sie sollten also unbedingt mit einer aktuellen Version starten. Parallel zur Drucklegung dieser Auflage wurde Version 2.2 frisch veröffentlicht.

Besuchen Sie <https://checkmk.com/de/download> und informieren Sie sich, welche Installationspakete gerade aktuell für Ihre Distribution vorhanden sind.

Wählen Sie für den Einstieg die *Checkmk Free Edition* und die letzte Release-Version aus dem Zweig 2.2 passend zu Ihrer Distribution. Laden Sie das jeweilige Installationspaket per `wget` direkt herunter oder kopieren Sie es anderweitig auf Ihren zukünftigen Monitoring-Server.

### 29.2.1 Installation von Checkmk unter openSUSE

Auch wenn Sie eine freie Community-Version wie beispielsweise openSUSE 15.4 einsetzen, sollten Sie problemlos auf die angebotenen SLES-Pakete zurückgreifen können:

```
root@linux# wget https://download.checkmk.com/checkmk/2.2.0p1/\
check-mk-cloud-2.2.0p1-sles15sp4-38.x86_64.rpm
root@linux# zypper install ./check-mk-cloud-2.2.0p1-sles15sp4-38.x86_64.rpm
```

**Listing 29.1** Installation von Checkmk unter Debian/Ubuntu

### 29.2.2 Installation von Checkmk unter Debian/Ubuntu

Für Debian finden Sie vorbereitete Pakete:

```
root@linux# wget https://download.checkmk.com/checkmk/2.2.0p1/\
check-mk-cloud-2.2.0p1_0.bullseye_amd64.deb
root@linux# apt install ./check-mk-cloud-2.2.0p1_0.bullseye_amd64.deb
```

**Listing 29.2** Installation von Checkmk unter Debian/Ubuntu

### 29.2.3 Installation von Checkmk unter CentOS

Auch unter CentOS gestaltet sich die Installation einfach – nutzen Sie `dnf`, um das Installationspaket zu installieren:

```
root@linux# wget https://download.checkmk.com/checkmk/2.2.0p1/\
check-mk-cloud-2.2.0p1-el9-38.x86_64.rpm
root@linux# rpm --install ./check-mk-cloud-2.2.0p1-el9-38.x86_64.rpm
```

**Listing 29.3** Installation von Checkmk unter CentOS

### 29.2.4 Die erste Kontrolle – klappt alles?

Hat alles geklappt, können Sie das Kommandozeilen-Tool `omd` aufrufen:

```
linux:~ # omd version
OMD - Open Monitoring Distribution Version 2.2.0p1.cce
```

**Listing 29.4** Der erste Aufruf von »omd«

#### Der Installationsort von Checkmk

Rund um Checkmk gab es eine »Open Monitoring Distribution« (OMD). Auf allen möglichen Plattformen soll dabei ein einheitlicher Pfad `/omd` existieren. Für manuell installierte Linux-Systeme wäre `/opt/omd` jedoch sauberer, sodass Checkmk bei der Installation kurzerhand einen Symlink von `/omd` nach `/opt/omd` setzt, sodass Sie stets auch direkt `/omd` verwenden können.



## 29.3 Einrichtung der ersten Monitoring-Instanz

Auch wenn Sie Checkmk nur für sich allein verwenden möchten, müssen Sie als allerersten Schritt eine eigene Instanz anlegen – und diese dann auch gleich starten:

#### Sie müssen immer eine Instanz einrichten

Sie können mit einer Checkmk-Installation problemlos mehrere Instanzen betreiben, die völlig unterschiedliche Rechner, Datacenter oder Kundengruppen überwachen. So könnten Sie beispielsweise Testumgebung und Produktivbetrieb trennen, um einfach einen möglichst übersichtlichen und eindeutigen Blick auf die tatsächliche Lage zu haben.

Ob Sie Ihre eigene Instanz wie hier in den Beispielen »mysite« oder »meineserver« oder »main« nennen, ist völlig egal. Es ist nur der Name, unter dem diese Instanz später läuft: Die Wahl des Namens beeinflusst den Speicherort (`/omd/sites/mysite`), die User-ID der jeweiligen Prozesse (`mysite`) und die Browser-URL, unter der diese Instanz aufgerufen wird (`http://monitoring.example.com/mysite`).



```
linux:~ # omd create mysite
Adding /opt/omd/sites/mysite/tmp to /etc/fstab.
Creating temporary filesystem /omd/sites/mysite/tmp...OK
Updating core configuration...
Generating configuration for core (type cmc)...Creating helper config...OK
OK
Restarting Apache...OK
Created new site mysite with version 2.2.0p1.cce.demo.
```

The site can be started with `omd start mysite`.

The default web UI is available at <http://linux.heinlein-support.de/mysite/>

The admin user for the web applications is **cmkadmin** with password: **sGCWU3Aj**  
 For command line administration of the site, log in with 'omd su mysite'.  
 After logging in, you can change the password for cmkadmin with  
 'htpasswd etc/htpasswd cmkadmin'.

```
linux:/omd # omd start mysite
Temporary filesystem already mounted
Starting mkeventd...OK
Starting liveproxyd...OK
Starting mknotifyd...OK
Starting rrdcached...OK
Starting cmc...OK
Starting apache...OK
Starting dcd...OK
Starting redis...OK
Initializing Crontab...OK
```

#### Listing 29.5 Die erste eigene Instanz unter Checkmk

Damit Sie sich später problemlos in das Webinterface einloggen können, müssen Sie sich das hier zufällig generierte Passwort für den Web-User **cmkadmin** merken – in diesem Beispiel lautet es **sGCWU3Aj**. OMD hat bereits einen Linux-User mit dem Namen Ihrer Instanz erzeugt: also *mysite*. Wenn Sie nun in Ihrer Konsole auf diesen User *mysite* wechseln, können Sie *omd* sehr bequem nutzen, um alle Administrationskommandos automatisch für Ihre Instanz ausführen zu lassen. Einen solchen Userwechsel könnte man über `su - mysite` vornehmen; *omd* bringt dafür jedoch gleich ein eigenes Kommando mit, wie Sie in Listing 29.6 sehen: `omd su mysite`. Ihre Monitoring-Instanz besteht aus dem Zusammenspiel einer Vielzahl unterschiedlicher Dienste, die jedoch bequem über das *omd*-Kommando verwaltet werden. Sie können später auch einzelne Dienste (neu) starten oder stoppen:

```
linux:/omd # omd su mysite
OMD[mysite]:~$ omd status
mkeventd: running
liveproxyd: running
mknotifyd: running
rrdcached: running
cmc: running
apache: running
dcd: running
redis: running
crontab: running

```

```
Overall state: running
OMD[mysite]:~$ omd reload apache
Reloading apache
```

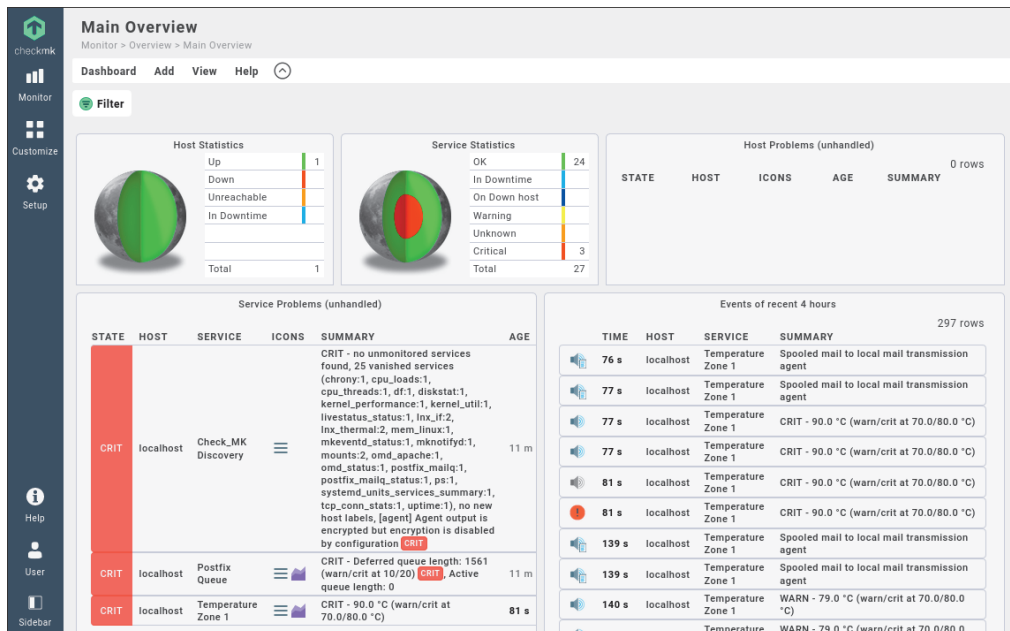
**Listing 29.6** Mit »omd« lassen sich die beteiligten Dienste verwalten.

### Weitere Details im Internet

Das Tool omd kann noch viel, viel mehr. Eine umfangreiche Anleitung zum Hinzufügen, Umbenennen, Stoppen, Löschen oder Backup Ihrer OMD-Instanzen finden Sie auf der Seite [https://docs.checkmk.com/latest/de/omd\\_basics.html](https://docs.checkmk.com/latest/de/omd_basics.html).

Es wird Zeit für den ersten Login. Rufen Sie im Browser <http://localhost/mysite> bzw. <http://<hostname>/mysite> auf, und loggen Sie sich mit dem Usernamen `cmkadmin` und dem von omd während der Installation ausgegebenen Passwort ein.

Nach dem Login finden Sie das (natürlich noch leere) Dashboard von Checkmk vor. In Abbildung 29.1 sehen Sie bereits die ersten eingerichteten Hosts. Zur besseren Lesbarkeit haben wir für alle folgenden Screenshots im Menüpunkt USER das INTERFACE THEME: DARK auf INTERFACE THEME: LIGHT umgestellt.



**Abbildung 29.1** Beim ersten Login wird Ihr Dashboard von Checkmk noch leer sein.



### Systemsprache Deutsch oder Englisch?

Im Menüpunkt `USER • EDIT PROFILE` können Sie in der Sektion `PERSONAL SETTINGS` Ihre Spracheinstellung von `ENGLISH` auf `DEUTSCH` umstellen. Allerdings beziehen sich viele sonstige Anleitungen und Screenshots oft auf die englische Systemsprache, sodass auch in diesem Buch alle Menüpunkte und Screenshots stets auf Englisch sein werden.

## 29.4 Server, Geräte und Dienste überwachen

Checkmk kennt sehr verschiedene Wege, über die Sie Linux- und Windows-Server, Netzwerkgeräte, Virtualisierungshosts, Drucker und vielleicht auch Ihre Kaffeemaschine überwachen können:

### ► Externes Überwachen von Netzwerkdiensten

Sie können mit Checkmk von außen auf das Zielobjekt zugreifen und beispielsweise mittels `ping` prüfen, ob es im Netzwerk antwortet, ob die richtigen Netzwerkports offen sind und ob Dienste richtig antworten. Das ist die grundlegende Überwachung für jedes Objekt, das Sie in Checkmk anlegen – und so können Sie auch Blackboxes wie Netzwerkkomponenten oder Drucker überwachen.

### ► Überwachung durch einen Checkmk-Agenten

Weitaus mehr Informationen erhalten Sie, wenn Sie auf den Servern den Checkmk-Agenten installieren. Das ist ein kleines Programm oder Skript, das vom Checkmk-Server von außen angesprochen und gestartet wird. Es sammelt lokal auf dem überwachten Host Informationen und kann so auch Prozessliste, Festplattenfüllstand und vieles mehr überwachen, was von außen nicht sichtbar ist. Der große Vorteil von Checkmk: Es kann auch dynamisch Entscheidungen treffen, was genau überwacht wird. Je nach Festplattenpartitionierung müssen auf jedem Server ja ganz unterschiedliche Bereiche des Dateisystems auf ihren Füllstand hin kontrolliert werden.

### ► Verwendung eigener Checks

Der Checkmk-Agent kann darüber hinaus selbst geschriebene Checks ausführen und bietet auch die Ausführung von klassischen Nagios-Plug-ins via `MRPE an`, dem Pendant zum `NRPE` von Nagios.

### ► Zugriff mittels SNMP

SNMP-Geräte kann Checkmk ebenfalls überwachen. Deren Einrichtung sprengt aber die Ausrichtung dieses Buches. Insbesondere ist es nicht ratsam, für die Überwachung von Linux- oder Windows-Systemen den dort mitgelieferten SNMP-Agenten zu nutzen. Der Checkmk-Agent kann mehr.

## 29.5 Installation des Checkmk-Agenten

Unter **SETUP • AGENTS** können Sie sich im Webinterface Checkmk die Installationspakete des Agenten für zahlreiche Betriebssysteme herunterladen, bequemerweise auch gleich als DEB- oder RPM-Paket (siehe Abbildung 29.2). Kopieren Sie diesen Agenten auf den zu überwachen- den Host, und installieren Sie ihn dort. Da das Paket nicht aus einer offiziellen Paketquelle stammt, werden Sie gegebenenfalls im Installationsprozess bestätigen müssen, dass Sie eine entsprechende Warnung ignorieren und es unsigned installieren möchten.

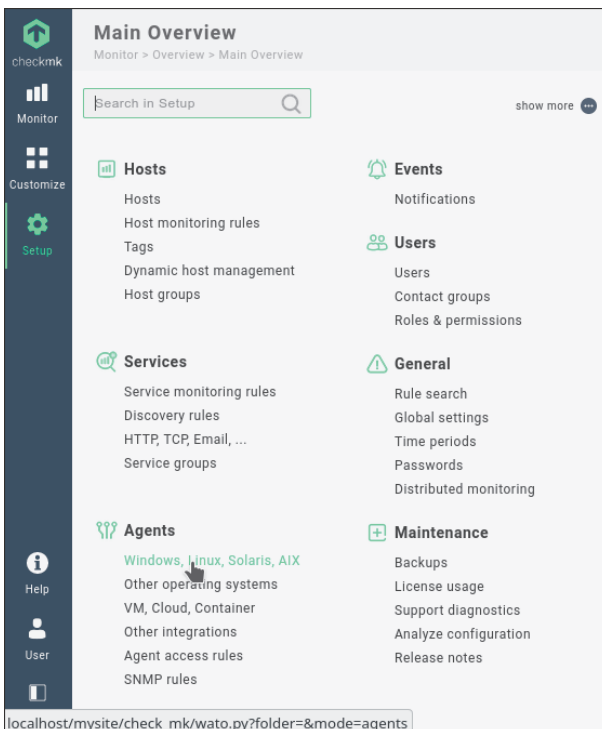
- ▶ Debian/Ubuntu:
 

```
apt install /path/to/check-mk-agent_2.2.0-f86cf10c14caa191_noarch.deb
```
- ▶ SUSE:
 

```
zypper install /path/to/check-mk-agent-2.2.0-f86cf10c14caa191.noarch.rpm
```
- ▶ CentOS:
 

```
dnf install /path/to/check-mk-agent-2.2.0-f86cf10c14caa191.noarch.rpm
```

Der Checkmk-Agenten wird als Dienst an Port 6556 installiert. Auf älteren Systemen wird dafür auf `xinetd` zurückgegriffen, auf neueren Systemen geschieht dies über `systemd`.



**Abbildung 29.2** Checkmk stellt über sein Webinterface auch Downloadquellen für den Agenten bereit.

Sie können die Funktionsfähigkeit Ihres Agenten schnell und einfach prüfen: Wenn Sie mittels `telnet localhost 6556` eine Verbindung zu diesem Port aufbauen, sollte er einen ganzen Stapel an Monitoring-Informationen liefern. Diese werden später vom Checkmk-Server erfasst und ausgewertet.

## 29.6 Anlegen eines Hosts

Es ist an der Zeit, Ihren Server ins Monitoring einzubinden. Gehen Sie auf den Menüpunkt **SETUP • HOSTS • ADD HOST**. Idealerweise hat Ihr Host einen im DNS sauber eingerichteten Hostnamen, sodass es reicht, hier den FQDN-Hostnamen einzutragen. Nur wenn Ihr Host nicht richtig im DNS erreichbar ist, müssen Sie unter **IPV4 ADDRESS** manuell eine IP-Adresse eintragen. Die weiteren Optionen können Sie zunächst unverändert lassen, per Default wird Checkmk den Host mithilfe des Checkmk-Agenten überprüfen.

Nach einem weiteren Klick auf **SAVE & GO TO SERVICE CONFIGURATION** wird Checkmk eine automatische Analyse des neuen Hosts vornehmen – und dafür auch den Checkmk-Agenten des Hosts kontaktieren. Klappt alles, wird sich Checkmk nach wenigen Sekunden mit einem bereits überraschend vollständigen Set an Service-Checks zurückmelden, die ab sofort überwacht werden können. Können – aber noch nicht werden!

Übernehmen Sie alle vorgeschlagenen Möglichkeiten, indem Sie auf der Seite durch einen Klick auf **ADD MISSING, REMOVE VANISHED** die vorgeschlagenen Service-Checks aktivieren.

**Activate pending changes**  
Setup > Activate pending changes 4 changes

Changes Related View Help

Currently there are 4 changes to activate.

**Activation status**

| ACTIONS | SITE              | STATUS | VERSION | CHANGES | PROGRESS | DETAILS                  |
|---------|-------------------|--------|---------|---------|----------|--------------------------|
|         | Local site mysite | online | 2.0.0b4 | 4       |          | Has never been activated |

**Pending changes** 4 rows

| TIME                | USER     | CHANGE                                                         | AFFECTED SITES |
|---------------------|----------|----------------------------------------------------------------|----------------|
| 2021-01-24 11:06:26 | cmkadmin | Modified host localhost.                                       | All sites      |
| 2021-01-24 11:04:54 | cmkadmin | Saved check configuration of host 'localhost' with 24 services | All sites      |
| 2021-01-24 11:02:42 | cmkadmin | Modified host localhost.                                       | All sites      |
| 2021-01-24 09:37:05 | cmkadmin | Created new host localhost.                                    | All sites      |

Abbildung 29.3 Änderungen wirken erst, wenn sie im laufenden Checkmk aktiviert werden.



Ihr Host ist nun inventarisiert und Checkmk weiß, welche Services zu überwachen sind – doch ist das nur graue Theorie, solange der permanent im Hintergrund laufende Monitoring-Kern diese Änderung noch nicht übernommen hat. Checkmk warnt Sie über ein (leider sehr) kleines gelbes Ausrufungszeichen rechts oben, dass es noch nicht übernommene Änderungen (*Changes*) gibt. Klicken Sie dort, und aktivieren Sie die anhängigen Änderungen (siehe Abbildung 29.3). Über den weiteren Menüpunkt **CHANGES • ACTIVATE ON AFFECTED SITES** vollenden Sie Ihr Werk. Das entspricht einem `systemctl reload` bei anderen Systemdiensten.

Erst jetzt werden Sie Ihren neu angelegten Host mit Alarmen im Dashboard sehen oder können ihn manuell über die Suchmaske im Menü **MONITOR** anzeigen lassen.

## 29.7 Betriebs- und Fehlerzustände von Host und Services im Überblick

29

Checkmk verwendet – ganz analog zu anderen Monitoring-Systemen wie Nagios – natürlich verschiedene Zustandsbezeichnungen für Dienste und Services. Sie werden darauf im laufenden Betrieb bei Ihrer Arbeit mit Checkmk immer wieder stoßen, darum seien sie hier einmal übersichtlich zusammengefasst. Für einen Host existieren folgende Zustände:

- ▶ **UP**  
Der Host ist normal erreichbar.
- ▶ **DOWN**  
Ein Host ist »down«, wenn er nicht erreichbar ist, das heißt: wenn das beim Objekt definierte `check_command` fehlschlägt.
- ▶ **UNREACH**  
Ein Host ist »unreachable«, wenn er selbst und gleichzeitig die als *parents* eingetragenen Hosts nicht erreichbar sind.<sup>1</sup>

Ein Service hingegen kann folgende Zustände haben:

- ▶ **OK**  
Das Plug-in liefert »OK« (Returncode 0) zurück – alles ist in Ordnung bzw. wie erwartet.
- ▶ **WARN**  
Anhand der Messwerte (und übergebenen Schwellenwerte) stellt das Plug-in fest, dass der Service zwar grundlegende Funktionen zur Verfügung stellt, aber nicht hundertpro-

<sup>1</sup> Für die Einrichtung von Parents und anderen Abhängigkeiten fehlt uns in diesem Buch leider der Platz. *Parents* sind Systeme, von denen ein überwachter Host abhängt – beispielsweise (s)ein Switch oder das Gateway. Fällt eine solche zentrale Komponente aus, sollten nicht gleich alle davon abhängigen Hosts als »down« alarmiert werden. Zum einen könnten diese ja durchaus noch laufen, zum anderen ist es in solchen Großlagen umso wichtiger, schnell erkennen zu können, welche Systeme die eigentliche Ursache für alle Alarme sind und was gegebenenfalls nur Folgesymptome sind.

zünftig funktioniert. Alternativ stellt der Check fest, dass der erste Schwellenwert über- bzw. unterschritten wurde. Der Check beendet sich dann mit dem Returncode 1.

▶ **CRIT**

Der Service ist nicht mehr verfügbar, oder der zweite Schwellenwert wurde über- oder unterschritten. Der Check beendet sich in diesem Fall mit dem Returncode 2.

▶ **UNKNOWN**

UNKNOWN ist der Zustand, in dem das Monitoring-System nicht mehr automatisch entscheiden kann, in welchem Zustand das überwachte Objekt ist. Das kann ein Fehler in den Daten sein, ein Bug in der Implementierung – oder es ist einfach das überwachte Objekt weggefallen. So tritt »UNKNOWN« z. B. auch bei ausgehängten Dateisystemen auf. Diese sind dann plötzlich nicht mehr in der Agentenausgabe enthalten und der zuständige Service-Check kann nur noch zum Ergebnis »UNKNOWN« kommen. Am Ende heißt es eigentlich immer, dass menschliche Intelligenz gefragt ist, um die Situation aufzulösen. Das Plug-in beendet sich mit einem Returncode größer oder gleich 3.

Daneben gibt es noch zwei Ereignisse, die sowohl einen Host als auch einen Service betreffen können und die in Logmeldungen oder in der Historie auftauchen können:

▶ **Flapping**

Der Host/Service springt oft zwischen OK und Nicht-OK hin und her. Um zu viele Benachrichtigungen zu unterdrücken, wird er als »flapping« markiert und eine Benachrichtigung darüber versandt. Weitere Benachrichtigungen werden unterdrückt. Erst wenn der Host oder der Service wieder zur Ruhe gekommen ist und einen stabilen Zustand zeigt, wird der Zustand »flapping« aufgehoben.

▶ **Scheduled downtime**

Über das Web-Frontend wurde eine Downtime eingetragen, das heißt, der Host/Service ist zum Beispiel für Wartungsarbeiten abgeschaltet worden und Alarme sind dementsprechend deaktiviert.

## 29.8 Konfiguration durch Regelsätze

Die Konfiguration von Checkmk erfolgt grundsätzlich anders als in vielen anderen Software-Systemen: Sie besteht aus einer Vielzahl von Regelsätzen, deren Filterbedingungen darüber entscheiden, ob sie angewendet werden sollen. In den Regelbedingungen lassen sich unglaublich viele Bedingungen prüfen. Dazu gehören beispielsweise (Regexp-)Pattern zum Hostnamen oder die Auswertung von Merkmalen wie *Host-Tags* oder *Label*, die Sie einem Host gegeben haben.

Denn selten ist eine Monitoring-Konfiguration einfach, linear und vorhersehbar. Schon die Entscheidung, ab welchem Füllstand einer Festplatte ein Alarm ausgelöst wird, kann höchst unterschiedlich ausfallen.

Für ein und denselben Check könnte als Szenario Folgendes im Raum stehen:

▶ **Virtuelle minimale Festplatte**

Handelt es sich um eine normale virtuelle Maschine mit sehr kleiner Betriebssystemplatte, so wird diese natürlich sehr schnell bzw. permanent gut gefüllt sein, sodass man Grenzwerte nach oben verschieben muss.

▶ **Physische große Festplatte**

Handelt es sich um eine reale physische Hardware, ist vielleicht eine völlig überdimensionierte echte Festplatte eingebaut. Trotzdem möchten Sie bei Anomalien und bei einem ungewöhnlichen Wachstum sehr frühzeitig informiert werden.

▶ **Tbyte-starker Fileserver**

Auch kann es sich bei dem fraglichen Host vielleicht um einen großen Datenspeicher mit einer viele TByte großen Datenpartition handeln, bei dem Sie schon dann einen Alarm bekommen möchten, wenn »nur noch« 1 Tbyte freier Festplattenplatz verfügbar ist.

▶ **Testsystem mit minimalen Ressourcen**

Und schlussendlich kann noch mal alles ganz anders sein, wenn das fragliche System in einer Testumgebung steht. Denn auch dann, wenn hier nichts unbeobachtet bleiben soll, muss eine übervolle Festplatte in keinem Fall einen 24/7-Alarm auslösen, bei dem der Administrator nachts geweckt wird.

Also: Fast jeder Aspekt der Konfiguration von Hosts und Services wird mit Checkmk als Regel in Regelsätzen abgebildet. Jeder Regelsatz ist dabei für genau einen Aspekt zuständig und wird von oben nach unten abgearbeitet, um die aktuelle Konfiguration zu erstellen. Die einzelnen Regeln werden nach dem Prinzip »Wenn, dann« ausgewertet. Das heißt, in jeder Regel ist eine Kombination von Bedingungen definiert sowie eine Aktion, die erfolgen soll, wenn die Bedingungen zutreffen. Welche Bedingungen möglich sind, ist natürlich je nach Check auch etwas unterschiedlich.

### 29.8.1 Arbeiten in Host-Ordnern

Im Setup-Bereich für Hosts haben Sie die Möglichkeit, Ordner und Unterordner anzulegen und jeden Host in einem passenden Ordner zu erstellen. Für die Ordner selbst können Sie bereits Merkmale konfigurieren, die sich dann auf die enthaltenen Hosts vererben. Das sind z. B. die Einstellungen zum Zugriff auf den Checkmk-Agenten oder via SNMP (inklusive Credentials) oder die Adressfamilie (IPv4 oder IPv6). Auch selbst definierte Merkmale (*Host Tags*) können so vererbt werden. Oft werden die Ordner anhand von Standorten oder technischen Merkmalen (Linux, Windows, Netzwerkkomponente) erstellt. Es können aber auch Ordner für einzelne Teams erstellt werden, die dann über die Kontaktgruppen mit eigenen Rechten ausgestattet werden. Wenn Sie im weiteren Kapitel gleich Checkmk-Regeln erstellen werden, können Sie diese wahlweise im Hauptverzeichnis *Main directory* oder in einem Unterordner speichern, sodass sie nur für alle Hosts in und unterhalb dieses Ordners angewandt werden.



### Regelsätze zwischen Ordnern verschieben

Wenn Sie eine Regel in einem Unterordner speichern, ist das de facto das Gleiche, als wenn Sie innerhalb des Regelsatzes angeben, dass diese Regel auf alle Hosts oder nur auf Hosts in diesem bestimmten Ordner angewandt werden soll. Wenn Sie die Regel in einem Ordner speichern, erzeugt Checkmk automatisch eine entsprechende Filterbedingung in dieser Regel. Und so können Sie später auch Regelsätze »verschieben«, indem Sie sie editieren und im Regelsatz unter **CONDITIONS** die Einstellung **FOLDER** ändern.

Einerseits könnten Sie jetzt allen Hosts in einem Ordner ein bestimmtes Attribut durch den Ordner vererben lassen: *Testsystem* oder *Virtuelles System*. In der Praxis zeigt sich in größeren Setups jedoch schnell, dass Hosts auch komplexer in Ordner sortiert werden sollen (Kundenmaschinen, interne IT) und es zudem schnell passiert, dass beim Umsortieren von Hosts in andere Ordner ganze Regelsätze plötzlich nicht mehr zur Anwendung kommen. Zudem können viele Regelsätze nicht einfach so verschoben werden, sondern müssen alle individuell editiert und neu gespeichert werden.



Es empfiehlt sich darum, in der Regel alle Regelsätze stets in *Main* abzuspeichern, selbst wenn sie nur für einen Ordner gelten sollen. Stattdessen weisen Sie dem jeweiligen Host lieber über seinen Ordner zunächst ein Host-Tag zu. Anschließend müssen Sie in den Regelsätzen nicht mehr auf den Speicherort der Regel abstellen, sondern können das Host-Tag als Regelbedingung verwenden.

Dieser indirekte Weg ist zwar zunächst aufwendiger und »einmal über die Bande gespielt«, doch können Sie so das gleiche Host-Tag in mehreren Ordnern setzen lassen und bleiben deutlich flexibler in Ihren Möglichkeiten. Auch fällt es später leichter, einem zusätzlichen neuen Ordner ebenfalls dieses Tag zuzuweisen, als wenn Sie eine Vielzahl von Regelsätzen duplizieren oder manuell editieren müssen, um sie in diesen neuen Ordner zu verschieben oder zu kopieren. Und so sieht das in der Praxis aus:

1. Legen Sie über das Menü **SETUP • HOSTS • ADD SUBFOLDER** neue Ordner an, zum Beispiel die zwei Ordner *PROD* und *TEST*.
2. Wechseln Sie in den *TEST*-Ordner, und gehen Sie über das Menü **FOLDER • PROPERTIES** in dessen Einstellungen.
3. Im Bereich **ATTRIBUTES** können Sie die Kritikalität auf **TEST SYSTEM** setzen.
4. Wechseln Sie zurück auf das *Main Directory*. In den Icons eines jeden dort gespeicherten Hosts haben Sie ein Icon mit Pfeil, um den Host in einen anderen Ordner zu verschieben. Verschieben Sie also Ihren Test-Host in den Ordner *TEST*.
5. Kontrollieren Sie das Ergebnis: Wechseln Sie erneut in den *TEST*-Ordner, und rufen Sie die Eigenschaften des Hosts auf. In seinen Attributen finden Sie nun unter **CRITICALITY** den Wert **TEST SYSTEM** nebst dem Vermerk, dass dieses vom Ordner *TEST* vererbt wurde.

Auch wenn die Kritikalität des Hosts nun TEST SYSTEM ist, bedeutet dies in der Praxis noch gar nichts. Es ist nur ein Merkmal des dort gespeicherten Hosts, nichts weiter. Doch dieses Merkmal können Sie nun nicht nur bequem als Suchkriterium in Checkmk verwenden, sondern auch bei der Definition von Regelsätzen als Bedingung abfragen, um Aktionen damit zu verbinden.

### 29.8.2 Keine Alarme für Testsysteme

Ein schönes Beispiel dafür ist, dass alle Hosts, die als TEST SYSTEM klassifiziert sind, kurzerhand keine Alarme auslösen sollen. Das hat nichts mit dem eigentlichen Check zu tun, sondern steht am Ende der Arbeitskette, wenn es darum geht, über die Konsequenzen eines Alarms zu entscheiden. Checkmk kennt dafür besagtes Flag namens *Criticality*.

1. Gehen Sie auf den Menüpunkt SETUP • SERVICE MONITORING RULES.
2. Beim ersten Aufruf sehen Sie vermutlich eine ganze Wand von Parametern und Möglichkeiten, doch sind diese in verschiedene Bereiche gruppiert, die Sie auf- und zuklappen können, sodass es schnell übersichtlich wird, wenn Sie ein paar Bereiche minimieren.
3. Im 5. Abschnitt, NOTIFICATIONS, finden Sie den Punkt ENABLE/DISABLE NOTIFICATIONS FOR SERVICES.
4. Wählen Sie ihn aus, und erzeugen Sie über den grünen Button eine neue Regel im Ordner *Main directory*.
5. Geben Sie Ihrer Regel einen Namen, z. B. »Keine Alarme bei Testsystemen«, und setzen Sie gegebenenfalls einen für Sie hilfreichen Kommentar.
6. Wählen Sie dann in der Sektion ENABLE/DISABLE NOTIFICATIONS FOR SERVICES das Ergebnis, dass diese Regel DISABLE SERVICE NOTIFICATIONS bewirken soll, wenn die gleich definierten Regelbedingungen zutreffen.
7. Nun können Sie im Block CONDITIONS festlegen, wann diese Aktion ausgelöst wird. Da wir Alarme unterdrücken wollen, wenn es sich um ein Testsystem handelt, fügen wir im Bereich HOST-TAGS mittels ADD TAG CONDITION die Bedingung hinzu, dass CRITICALITY den Wert TEST SYSTEM haben muss.

Allerdings unterscheidet Checkmk konsequent zwischen *Services* und *Hosts* – ein Überbleibsel seines Nagios-Ursprungs. Darum existiert bis jetzt nur die Definition, dass für Services auf Testsystemen die Alarmierung abgeschaltet wird. Das hat jedoch nichts damit zu tun, wie sich Checkmk verhält, wenn der gesamte Host ausgefallen ist. Erzeugen Sie darum ganz analog eine identische Regel für alle Benachrichtigungen der Testsystem-Hosts:

1. Gehen Sie auf den Menüpunkt SETUP • HOST MONITORING RULES.
2. Wählen Sie im Bereich NOTIFICATIONS den Bereich ENABLE/DISABLE NOTIFICATIONS FOR HOSTS.

3. Erzeugen Sie eine neue Regel im *Main Directory*.
4. Setzen Sie die Aktion ENABLE/DISABLE NOTIFICATIONS FOR HOSTS und unter HOST TAGS die Filterbedingung CRITICALITY ist TESTSYSTEM.

### 29.8.3 Unterschiedliche Alarmschwellen bei Dateisystemen

Wenn wir uns anschauen, wie Sie sich unterschiedlich über Festplattenfüllstände benachrichtigen lassen können, lernen wir den Aufbau und die Logik von Checkmk gut kennen. Auch hier können Sie Regelsätze definieren und über die Kombination verschiedenster Bedingungen festlegen, welche Schwellenwerte im Einzelfall gelten sollen.

Am einfachsten lässt sich diese Regel definieren, wenn Sie nicht über die normale Menüstruktur, sondern über einen konkreten Check in die Konfiguration einsteigen:

1. Gehen Sie ins Checkmk-Dashboard, und wählen Sie Ihren bereits angelegten Test-Host aus, also beispielsweise *localhost*.
2. Unser Checkmk-Agent hat dort bereits einen Check »Filesystem /« eingerichtet, die Überwachung als solche ist bereits aktiv.
3. Im Aktionsmenü dieses Checks, dem »Hamburger-Menü« mit den drei horizontalen Strichen, können Sie PARAMETER FOR THIS SERVICE auswählen.
4. Neben dem Umstand, dass es sich bei diesem Check um einen »Inventurcheck« handelt, er also vom Checkmk-Agenten automatisch hinzugefügt wurde, sehen Sie darunter die derzeit aktiven Standardparameter – unter anderem: eine Warnung ab 80 % Füllstand, ein Alarm ab 90 % Füllstand.
5. Mit einem Klick auf FILESYSTEMS (USED SPACE AND GROWTH) können Sie wieder einmal eine neue Regel anlegen – achten Sie auch hier darauf, dass diese im *Main directory* angelegt wird.
6. Geben Sie Ihrer Regel wie immer einen aussagekräftigen Namen und bei Bedarf einen Kommentar.
7. Checkmk wird Ihnen hier stets nur die Parameter und Optionen anzeigen, die für diesen Check angewendet werden können.

Passen Sie gegebenenfalls folgende Optionen an:

- **Levels for filesystem**

Hier können Sie ganz allgemein einstellen, ab welchem Füllstand Warnungen oder Alarmierungen ausgegeben werden.

- **Magic Factor**

Der sogenannte *Magic Factor* hilft Ihnen, die eigentlich fixen Schwellenwerte an große oder kleine Dateisysteme anzupassen. Bei einem Wert unter »1« verschiebt sich die Alarmierungsgrenze umso weiter nach oben, je größer das Dateisystem ist.

Und natürlich verschiebt sich die Alarmierungsgrenze umso weiter auch nach unten, je kleiner Ihr Dateisystem ist (siehe Abbildung 29.4). Checkmk hilft Ihnen also, bei der Alarmierung nicht nur den rein prozentualen Füllstand, sondern auch einen gewissen absoluten Füllstand zu berücksichtigen. Auch wenn ein großes Dateisystem vielleicht zu 90 % voll ist, könnte es absolut gesehen noch immer zwei Terabyte freien Speicher haben.

- **Reference size for magic factor**

Doch was ist groß und was ist klein? Was ist der Normalzustand und was ist ungewöhnlich? Wählen Sie hier aus, auf welcher Standardfestplattengröße Ihre Angaben beruhen. Das ist sozusagen der »Nullpunkt« Ihres Magic Factors.

- **Minimum levels if using magic factor**

Liegt Ihre *Reference size* sehr hoch, könnten bei sehr kleinen Dateisystemen (z. B. root-Partitionen virtueller Server) schnell extrem niedrige Alarmgrenzen entstehen. Legen Sie hier also Minimalwerte fest, die nicht unterschritten werden dürfen.

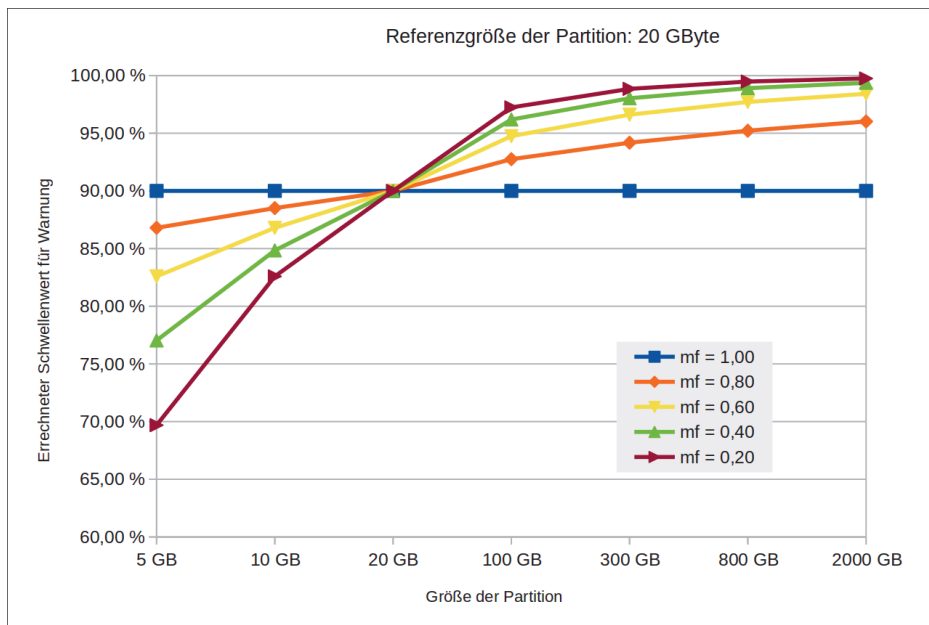


Abbildung 29.4 Die Auswirkungen des »Magic Factor«

- Nachdem Sie nun Optionen und Schwellenwerte des Checks definiert haben, könnten Sie im Block `CONDITIONS` noch genauer festlegen, ob diese Werte für alle Hosts gelten sollen oder nur dann, wenn bestimmte Filterkriterien zutreffen:
  - Über die Auswahl eines Speicherordners der Regel könnten Sie allgemein definieren, dass die Regel nur für Hosts in diesem Ordner gilt.

- Natürlich könnten Sie auch explizit Hostnamen oder Pattern von Hostnamen angeben, die erfüllt sein müssen.
- Und über Mount-Points könnten Sie festlegen, dass diese Regel nur für bestimmte Partitionsnamen gelten soll, beispielsweise für große /srv/data-Partitionen. Diese Angabe muss stets als RegExp-Pattern erfolgen. Möchten Sie also nur »/« überwachen, müssen Sie »/\$« angeben.
- Geben Sie mehrere Hostnamen an, werden diese natürlich mittels ODER verknüpft, während die sonstigen Bedingungen des Regelsatzes UND-verknüpft sind. Die Schwellenwerte gelten also nur dann, wenn der Hostname *und* der Mount-Point auf das gewählte Pattern passen.



#### Schneller lernen mit den Hinweistexten

Sie können auf den Checkmk-Seiten in der obersten Menüleiste über den Punkt **HELP • TOGGLE INLINE HELP** zusätzliche Erklärungstexte in den verschiedenen Bereichen einblenden lassen. Das macht die Seite vielleicht etwas unübersichtlicher, hilft Ihnen gerade zu Anfang aber oft sehr, zu verstehen, was gemeint ist.

### 29.8.4 Service Discovery Rules: Gezielt Prozesse überwachen

Nicht jeder Dienst lässt sich von außen überwachen – und selbst dann, wenn das möglich ist, ist es oft wichtig zu wissen, ob und gegebenenfalls wie oft ein Dienst auf dem überwachten Host läuft. Mit dem Checkmk-Agenten ist das kein Problem, denn er hat bei seiner Ausführung Zugriff auf die Prozessliste.

Doch etwas Hilfe benötigt Checkmk schon: Er muss wissen, welche Prozesse es zu überwachen gilt und wie er sie erkennt. Außerdem müssen Sie dem Check natürlich mitgeben, welche Werte als normal einzustufen und welche als Fehler anzusehen sind.

- ▶ Gehen Sie auf den Punkt **SETUP • SERVICE • DISCOVERY RULES**.
- ▶ Auf Linux-Systemen nutzen Sie dann **PROCESS DISCOVERY** und legen eine neue Regel im *Main directory* an.
- ▶ Entscheidend ist, dass Sie im Bereich **PROCESS MATCHING** das richtige Erkennungspattern angeben und Checkmk unter **LEVELS FOR PROCESS COUNT** mitteilen, dass er (genau) einmal laufen soll.
- ▶ Möchten Sie einschränken, dass ein Prozess nur auf bestimmten Hosts gesucht wird, können Sie unter **CONDITIONS** wie gewohnt auf Flags, Hostnamen oder Hostnamen-Pattern zurückgreifen.

Abbildung 29.5 zeigt die Regeldefinition für dieses Beispiel. Können Prozesse auch mehrfach vorhanden sein, sollten Sie die Schwellenwerte für *Warning above* bzw. *Critical above* natür-



lich höher setzen. So können Sie sich informieren lassen, wenn Ihr System zu sehr unter Last gerät. Und so aktivieren Sie diesen Check für Ihre Hosts:

- ▶ Gehen Sie auf die Detailansicht eines Hosts, und wählen Sie den Menüpunkt **HOST • SERVICE CONFIGURATION**.
- ▶ Ihr neuer Check wird im Block **UNDECIDED SERVICES (CURRENTLY NOT MONITORED)** gelistet sein. Durch einen Klick auf den grünen »+«-Button können Sie das Monitoring für diesen Host aktivieren.
- ▶ Bevor Sie das im Einzelfall entscheiden, können Sie Checkmk alternativ über **ADD MISSING, REMOVE VANISHED** oder **MONITOR UNDECIDED SERVICES** auch anweisen, alle derzeit gefundenen Prozesse zu überwachen bzw. die neuen hinzuzufügen.

**Edit rule: Process discovery**  
Setup > Services > Service discovery rules > Process discovery > Edit rule: Process discovery

Actions Related View Help

Save Abort Process discovery

**RULE PROPERTIES**

Description: Cron gestartet?

Comment: Falls Cron irgendwann mal nicht läuft, fällt das ggf. lange nicht auf!

Documentation URL:

Rule activation: do not apply this rule

**PROCESS DISCOVERY**

Process Name (required): Cron

Process Matching:  Exact name of the process without arguments

Executable: /usr/sbin/cron

Name of operating system user

Operating system control group information

Host Label

Default parameters for detected services

CPU rescale maximum load: 100% is all cores at full load

Levels for process count

Critical below 1 processes

Warning below 1 processes

Warning above 1 processes

Critical above 1 processes

Levels on total CPU utilization

Abbildung 29.5 »cron« wird überwacht.

Sie können Checkmk auch anweisen, den neuen Service auf *allen* überwachten Hosts erkennen zu lassen. Verwenden Sie dafür den Menüpunkt **SETUP • HOSTS**, und gehen Sie dann dort in den Menüpunkt **HOSTS • DISCOVER SERVICES**.



### Alle Änderungen müssen auch aktiviert werden!

Auch derartige Änderungen müssen Sie immer über `ACTIVATE PENDING CHANGES` in den laufenden Checkmk-Monitoring-Kern laden lassen. Das orangefarbene Ausrufungszeichen rechts oben informiert Sie über ausstehende Änderungen!

## 29.8.5 HTTP, TCP und E-Mail: Netzwerkdienste überwachen

Unsere bisherigen Checks waren darauf ausgelegt, den Host in seinem Zustand von »innen« zu überwachen, um Probleme frühzeitig zu erkennen. Genauso wichtig ist jedoch auch der Blick von außen auf das System, denn das ist die Sicht, die ein User hat: Lädt die Webseite? Lädt sie schnell? Und gegebenenfalls auch: Lädt sie den richtigen Inhalt oder zeigt sie einen Datenbankfehler? Checkmk kann darum nicht nur prüfen, ob dieser oder jener Netzwerkport offen und erreichbar ist, sondern auch, wie er reagiert und welche Antworten ein Netzwerkdienst liefert.

Und so richten wir einen HTTP-Check ein, der eine Webseite überprüft:

1. Im Menüpunkt `SETUP • SERVICES` finden Sie ein eigenes Untermenü für die Überwachung von `HTTP, TCP, EMAIL`.
2. Legen Sie dort im Bereich `CHECK HTTP SERVICE` im Ordner *Main directory* eine neue Regel an, und geben Sie ihr wie üblich einen Namen und bei Bedarf einen hilfreichen Kommentar.
3. Im Bereich `CHECK HTTP SERVICE` müssen Sie einen `SERVICE NAME` vergeben. Hostname, TCP-Port etc. müssen Sie nicht angeben, wenn diese nicht vom Standard abweichen. Als URI wird (ohne Anpassung) »/« überwacht.
4. Eventuell möchten Sie in den weiteren Parametern einstellen, in welcher *Response Time* die Seite antworten muss. Unter `FIXED STRING TO EXPECT IN THE CONTENT` können Sie Text angeben, den Ihre Webseite auf jeden Fall enthalten muss; alternativ sind auch Reg-Exp-Pattern möglich. »Impressum« wäre beispielsweise ein guter Begriff, wenn dieses Wort in jedem Footer Ihrer Webseite enthalten ist und Sie darüber auch indirekt erkennen können, dass der Webcontent erfolgreich aus der Datenbank geladen worden ist.
5. Gerade bei dieser Regel müssen Sie vermutlich unter `CONDITIONS` auch explizit den oder die Host(s) angeben, von dem bzw. denen dieser Webaufruf überwacht wird, sonst würde Checkmk plötzlich versuchen, von allen Ihren Servern im Netzwerk diese Webseite zu laden. Haben Sie mehrere systematisch benannte Webserver in einem Cluster, können Sie diese über ein RegExp-Pattern im Hostnamen auf einen Schlag definieren.

Vergessen Sie am Ende nicht den üblichen *Reload*, damit Ihre Änderungen auch im laufenden Checkmk-Dämon aktiv werden.



### Alle Änderungen müssen auch aktiviert werden!

Beginnen Sie einen Hostnamen mit »~«, so teilen Sie Checkmk mit, dass es sich im folgenden um ein RegExp-Pattern handelt. Für eine nummerierte Gruppe von Hosts in einem Webcluster wäre beispielsweise `~web[1-9].example.com` eine gute Definition mittels RegExp.

## 29.9 Notifications

Das schönste Alarm-Dashbboard nützt nichts, wenn der Bildschirm ausgeschaltet ist oder der Admin schläft. Alarmer müssen per Push-Nachricht zum Administrator gelangen, sei es per E-Mail, per SMS oder mit einer anderen Methode. Legen Sie darum für sich – und gegebenenfalls Ihre Kollegen – unter `SETUP•USER` eigene Nutzerkennungen an, und vergessen Sie nicht, dort auch eine E-Mail-Adresse zu hinterlegen. Dann müssen Sie auch nicht mehr mit dem Account `cmkadmin` arbeiten, denn das ist praktisch der `root`-Account von Checkmk mit den höchsten Rechten. Stellen Sie sodann bei Ihrem neuen Nutzer ein, für welche Hosts er Benachrichtigungen bekommen soll. Checkmk nutzt dafür sogenannte *Contact Groups*, die Sie Hosts und Nutzern zuweisen können. In unserem Fall soll der Account zunächst Benachrichtigungen von allen Hosts erhalten. Kreuzen Sie also in `CONTACT GROUPS` das Feld `EVERYTHING` an. Der neue Account benötigt ebenfalls eine Rolle, die ihm Berechtigungen in der Web-Oberfläche zuweist. Dafür gibt es drei vordefinierte Rollen:

- ▶ **Administrator**  
Darf alles.
- ▶ **Gast**  
Darf nur gucken, aber nichts ändern.
- ▶ **Normal Monitoring User**  
Darf Dinge ändern, soweit er Zugriffsrechte hat.

### 29.9.1 Anlegen weiterer Kontaktgruppen

Ist Ihr Kollegenkreis größer oder möchten Sie auch Kunden direkt vom Monitoring informieren lassen? Unter `SETUP•USERS•CONTACT GROUPS` können Sie jederzeit weitere Kontaktgruppen anlegen. Tragen Sie sodann User und auch Hosts in die gewünschten Kontaktgruppen ein, und kanalisieren Sie so, wer welche Alarmer bekommt und wer nicht.

Richten wir als Beispiel eine neue Regel ein, die nur die Postmaster unserer Mail-Relays informieren soll:

1. Wechseln Sie in das Menü `SETUP•USERS•CONTACT GROUPS`, wo bereits eine Gruppe `all/Everything` existiert.

2. Legen Sie eine Kontaktgruppe *Postmaster* an (oder *Webmaster*, wenn Ihnen das als Beispiel lieber ist), und geben Sie ihr einen (internen) Namen und einen Alias (für die Anzeige im Webinterface).
3. Anschließend können Sie in diesem Menü über den Button RULES Zuweisungsregeln für Hosts und Services erzeugen. Legen Sie im *Main directory* eine neue Host-Regel an, tragen Sie unter ASSIGNMENT OF HOSTS TO CONTACT GROUPS die neue Gruppe ein, und definieren Sie dann unter CONDITIONS wahlweise über HOST TAGS, HOST LABELS und natürlich EXPLICIT HOSTS die passenden Filterkriterien. Denken Sie dran: Auch RegExp-Pattern sind in EXPLICIT HOSTS möglich!
4. Legen Sie nun über SETUP • USERS • USERS Ihre neuen User an, und weisen Sie sie über CONTACT GROUPS der neuen Gruppe zu.



Betreiben Sie Kundenserver? Ganz analog zu dieser Anleitung können Sie auch für größere Kunden eine eigene Kontaktgruppe anlegen und die Alarme auf die Hostnamen ihrer Server limitieren.

### 29.9.2 Test der E-Mail-Zustellung

Checkmk übergibt Alarm-E-Mails an den lokalen Mailserver auf dem Monitoring-Server. Stellen Sie also sicher, dass Ihr lokaler MTA, also z. B. Postfix, korrekt konfiguriert ist und E-Mails ins Internet zustellen kann. In Kapitel 10, »Mailserver«, stellen wir Ihnen die Konfiguration eines Mailsystems vor. Ob alles einwandfrei funktioniert, können Sie über eine von Checkmk erzeugte Test-E-Mail prüfen. Rufen Sie dazu die Host-Ansicht Ihres Test-Hosts auf bzw. lassen Sie sich den Host über das Suchfeld im Menü MONITORING heraussuchen. Wählen Sie sodann den Menüpunkt COMMANDS • CUSTOM NOTIFICATIONS, und lassen Sie einen erzwungenen Test-Alarm an alle erzeugen.

Checkmk wird umgehend eine Alarm-Benachrichtigung per E-Mail versenden. Kommt diese bei Ihrer Mailadresse an, ist alles in Ordnung. Warten Sie vergebens, so prüfen Sie nochmals die im Checkmk-Account hinterlegte Mailadresse (siehe oben). Die Logfiles von Checkmk in `$OMD_ROOT/var/log` können hier sehr hilfreich sein, insbesondere das *notify.log*.

### 29.9.3 Alarmierung per SMS

Für eine Alarmierung per SMS gibt es mehrere Möglichkeiten, die wir hier leider nur kurz skizzieren können:

#### ► Internes SMS-Modem und *smstools*

ein an den Monitoring-Server angeschlossenes SMS-fähiges GSM-Modem, das von den *smstools* unterstützt wird. Die *smstools* müssen dafür außerhalb von Checkmk eingerichtet werden.

- ▶ **Internes GSM-Modem mit HTTP-API**  
ein GSM-Modem im Netzwerk mit HTTP-API, für das ein eigenes Notification-Plug-in erstellt werden muss.
- ▶ **Externes Mail2SMS-Gateway**  
ein Dienstleister, der ein Mail2SMS-Gateway anbietet. An diesen werden die Alarme per E-Mail geschickt. Dafür kann z. B. die Notification-Methode ASCIIMAIL verwendet werden.
- ▶ **Externes SMS-Gateway mit HTTP-API**  
ein Dienstleister, der eine HTTP-API zur Versendung von SMS anbietet. Für dieses Gateway muss ebenfalls ein eigenes Notification-Plug-in erstellt werden.

Grundsätzlich muss dann in `SETUP•EVENTS•NOTIFICATION CONFIGURATION` eine weitere Notification-Regel mit der entsprechenden Methode eingerichtet werden. In dieser Regel kann dann auch festgelegt werden, dass über diese Methode z. B. nur zu bestimmten Tageszeiten (`TIME PERIOD`) alarmiert werden soll, beispielsweise weil Mail-Alarme tagsüber ausreichen und SMS-Alarme unnötig sind.

#### 29.9.4 Wann wird ein Fehler zum HARD STATE?

Nicht immer, wenn ein Check mal fehlschlägt, ist gleich Hektik angesagt. Wenn ein System kurzzeitig mehr Last hat, sich aber nach wenigen Minuten wieder beruhigt hat, kann das ja völlig normal sein. Auch ein Dateisystem kann kurzzeitig mal voller oder leerer werden, zum Beispiel weil Logrotate Ihre Daten komprimiert und daher etwas zwischenspeichern muss. Auch auf einem Mailserver können einige Tausend aufgestaute Mails in der Mail-Queue absolut in Ordnung sein, zum Beispiel, weil jemand einen Massenversand gestartet hat. Bemerkenswert wird es ja nur, wenn sich dieser Peak nicht mehr zeitnah abbaut.

Checkmk unterscheidet darum zwischen einem *Soft State* und einem *Hard State*. Schlägt ein Check fehl, so wird dies auch als *Soft State* auf dem Dashboard angezeigt und in den Protokollen vermerkt, löst aber noch keine weiteren Aktionen wie beispielsweise eine Alarmierung aus. Erst wenn ein Check eine bestimmte Anzahl von Versuchen hintereinander durchgehend fehlschlägt, wird das Problem zum *Hard State* und führt zur Benachrichtigung des Administrators.

Sie können dabei für jeden einzelnen Check individuell festlegen, wie schnell oder geduldig er Fehler tolerieren soll. Darum finden sich diese Einstellungen auch in `PARAMETERS FOR THIS SERVICE`. Schauen wir uns das Ganze am Beispiel der CPU-Utilization eines Systems an:

1. Gehen Sie ins Dashboard, und lassen Sie sich die Services Ihres überwachten Hosts anzeigen.
2. Suchen Sie die Zeile des Service `CPU LOAD`, und klicken Sie in dessen Action-Menü auf `PARAMETERS FOR THIS SERVICE`.

3. Sie sehen: Standardmäßig erzeugt dieser Check einen Fehler, wenn die CPU LOAD über 5.00 (Warnung) bzw. 10.00 (Critical) angestiegen ist.
4. In den SERVICE MONITORING RULES gibt es allerdings den Wert MAXIMUM NUMBER OF CHECK ATTEMPTS FOR SERVICE, der standardmäßig auf 1 gesetzt ist. Das würde bedeuten: Jeder Peak in der Load führt zum Alarm. Das kann sinnvoll sein – muss aber nicht.
5. Mit einem Klick auf MAXIMUM NUMBER OF CHECK ATTEMPTS FOR SERVICE können Sie eine neue Regel erstellen, die Sie wie gewohnt im Ordner *Main directory* speichern sollten.
6. Wenn der Check jede Minute ausgeführt wird (so der Standard) und Sie hier die Maximalanzahl auf 5 setzen, darf Ihr System 4 Minuten lang im Peak sein, denn erst ab dem 5. Fehler in Folge wird das als echtes Problem gewertet.

### 29.9.5 Definieren von Notification Periods

Nicht jedes Problem ist zu jeder Uhrzeit relevant – und spätestens dann, wenn Sie auf Alarmierungen per SMS umgestellt haben, möchten Sie nicht nachts wegen zeitunkritischer Probleme 24/7 aus dem Schlaf gerissen werden. Sind weitere *Periods* definiert, können Sie für einen Check auch eine ganz individuelle Alarmierungsperiode setzen lassen:

- ▶ Definieren Sie als allererstes über den Menüpunkt **SETUP • GENERAL • TIME PERIODS** eine neue Zeitperiode mit Ihren individuellen Tagen, Uhrzeiten und gegebenenfalls auch Feiertagen.
- ▶ Anschließend können Sie sich einen bereits überwachten Host suchen, auf dem der gewünschte Check aktiv ist.
- ▶ Klicken Sie im Action-Menü des Service auf **PARAMETERS FOR THIS SERVICE**.
- ▶ Mit einem Klick auf **NOTIFICATION PERIOD FOR SERVICES** können Sie wieder wie gewohnt eine neue Regel im *Main directory* speichern.
- ▶ Setzen Sie die **NOTIFICATION PERIOD** auf Ihre neue Zeitperiode.
- ▶ Überlegen Sie, ob diese Alarmierungsregel nur für diesen Check auf diesem Host, oder global gelten soll: Tragen Sie also gegebenenfalls Hosts unter **EXPLICIT HOSTS** ein oder deaktivieren Sie diesen Regelbestandteil.

## 29.10 Alarmer managen

Löst Checkmk für einen bestimmten Check den Zustand »WARN« (Warning) oder »CRIT« (Critical) aus, wird der Service Check auf dem Checkmk-Dashboard deutlich sichtbar angezeigt. Außerdem wird er nun in kürzeren Zeitabständen überwacht, um möglichst bald Entwarnung geben zu können – oder auch weiter Alarm schlagen zu können, falls sich ein »WARN« zu einem »CRIT« verschlechtert. Das wird am Ende auch dazu führen, dass die beteiligten

Administratoren regelmäßig weitere Alarmmeldungen per E-Mail zum gleichen Problem bekommen.

Auch bei der Arbeit im Team ist es stets wichtig zu wissen, ob ein akut auftretendes Problem bereits von jemandem bearbeitet wird oder nicht. Um einen Alarm zu bearbeiten und zu kennzeichnen, können Sie einen Host (mitsamt aller Services) in Wartungszeit setzen oder einen einzelnen Alarm quittieren:

► **Scheduled Downtime** (»Wartungszeiten ansetzen«)

Dieser Zustand sagt aus, dass der Service geplant nicht mehr »OK« sein wird, zum Beispiel aufgrund von durchgeführten Wartungsarbeiten. Ein Ausfall des Service ist also entweder vorbereitet, geplant oder vorübergehend akzeptiert, und das Problem kommt vielleicht durch vorbereitete Loadbalancing-Maßnahmen äußerlich gar nicht zum Tragen. Nutzen Sie diesen Zustand also, um auszusagen: »Der Check ist zwar nicht okay, aber wir haben kein ernsthaftes Problem, um daß wir uns gerade kümmern müssen.«

► **Acknowledgement** (»Probleme quittieren«)

Ein *Acknowledgement* (ACK) sagt aus, dass der verantwortliche Administrator das Problem zur Kenntnis genommen hat. Das Problem besteht also weiterhin und ist akut und unter Umständen auch für Anwender sichtbar, es besteht nur kein Grund mehr, weiter Alarm zu schlagen. Oder mit anderen Worten: Das Feuer brennt noch, aber die Alarmsirene wurde ausgeschaltet, um die Nerven zu schonen und den Blick auf das Wesentliche richten zu können.

Die Quittierung muss immer mit einem aussagekräftigen Kommentar versehen werden. Hier kann z. B. eine Ticket-ID, der Name des nun bearbeitenden Administrators oder Ähnliches eingetragen werden.

Klicken Sie im Fall der Fälle also auf einen Alarm, und gehen Sie dadurch auf dessen Detailansicht. In der Menüleiste finden sich dann unter COMMANDS die Menüpunkte ACKNOWLEDGE PROBLEMS bzw. SCHEDULE DOWNTIMES. Gerade wenn Sie mit Kollegen parallel arbeiten, empfiehlt es sich, bei einem ACK oder einer Downtime einen passenden Kommentartext zu hinterlegen, damit alle im Team Bescheid wissen. Auch zu einem späteren Zeitpunkt lässt sich dann stets noch nachvollziehen, was die Ursache war und welche Abhilfe damals geschaffen wurde.

### Quittierung zeitlich limitieren!

Setzen Sie eine Quittierung idealerweise zeitlich beschränkt. Vielleicht kommt irgendetwas dazwischen und ein nicht direkt spürbares Problem wird letzten Endes gar nicht gelöst. Im Falle einer unlimitiert gesetzten Quittierung würde Checkmk dann quasi »nie wieder« das nach wie vor bestehende Problem melden, solange sich der Status nicht ändert. Das ist ein häufiges Problem beispielsweise bei der Überwachung des Festplattenfüllstandes!



Gerät das Problem »Dateisystem ist > 95 % voll« doch in Vergessenheit, wird es einige Tage oder Wochen später zum tatsächlichen Ausfall kommen, wenn der fatale Zustand »Dateisystem ist definitiv zu 100 % voll« erreicht ist. Allerdings: Für dieses Feature müssen Sie die lizenzierte *Checkmk Enterprise Edition* einsetzen.

### 29.10.1 Die mächtige Suche von Checkmk

Je komplexer Ihr Setup wird, umso weniger wird es ausreichen, über das normale Suchfeld im MONITOR-Menü eine schnelle Suche durchzuführen. Über den Menüpunkt MONITOR • HOST SEARCH öffnen Sie am rechten Seitenrand die mächtige Suchmaske aus Abbildung 29.6.

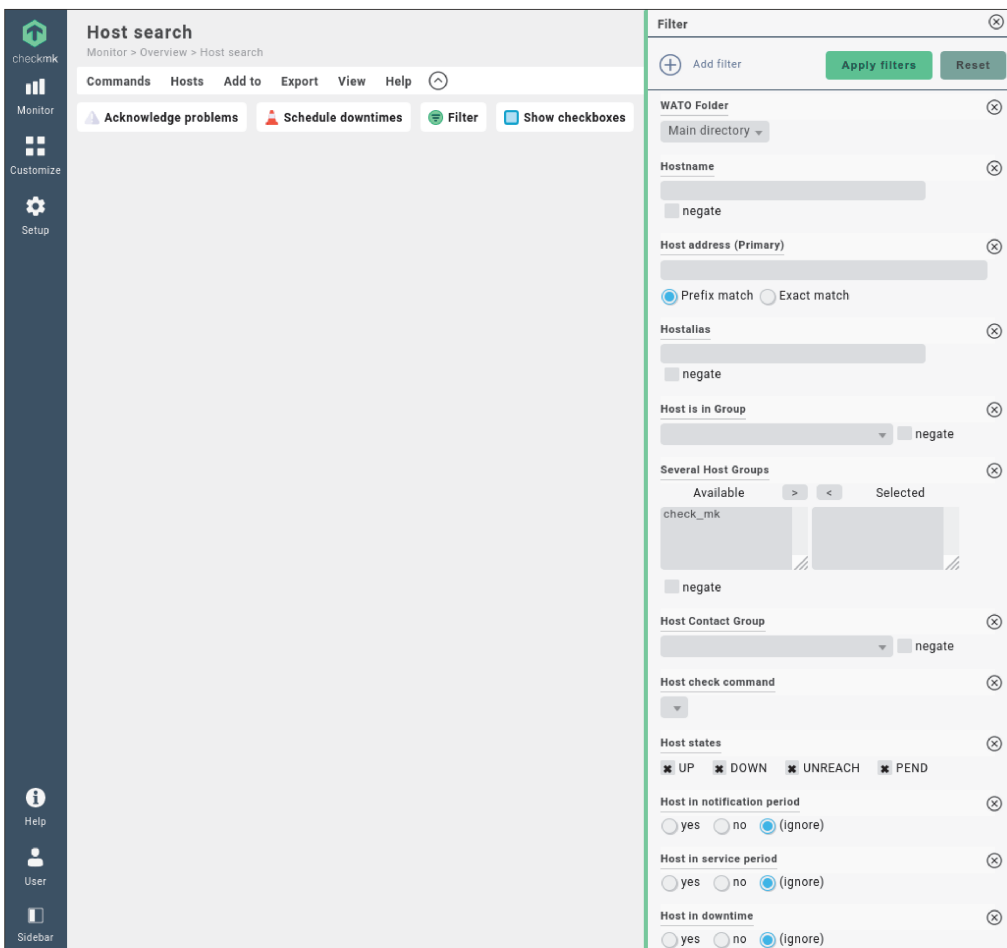


Abbildung 29.6 Die mächtige Host-Suche macht alles möglich.



Sie können dort nach allen möglichen typischen Hostmerkmalen suchen, auch nach Ihren Host-Tags. Klicken Sie sich beispielsweise als Filterkriterium das Merkmal CRITICALITY = TEST SYSTEM so zusammen wie in Abbildung 29.7.

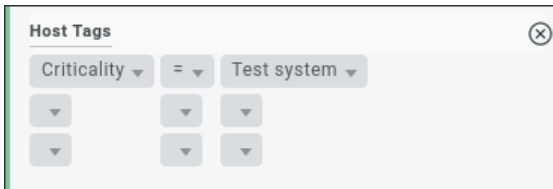


Abbildung 29.7 Auch nach Host-Tags lässt sich suchen.

Sie können aber auch ganz dynamisch nach aktuellen Zuständen in Ihrem Netzwerk suchen. Auch die Liste aller Hosts, die derzeit »DOWN« oder »UNREACH« sind, kann ja stets interessant sein. Beachten Sie auch hier wieder die stete Trennung, die Checkmk zwischen »Hosts« und »Service« vornimmt. Unter MONITOR • SERVICE SEARCH finden Sie eine ganz analoge Suchmaske für alle Ihre Checks.

Sie können eine Suche auch bequem als Bookmark im Browser speichern und sie so sehr schnell und einfach immer wieder aufrufen.



## 29.11 Weitere Fähigkeiten von Checkmk

Checkmk kann noch viel mehr, sehr viel mehr, doch dies ist Stoff und Thema für ein komplett eigenes Buch oder gar für eine Schulungswoche. Mittlerweile verfügt Checkmk jedoch auch über eine gut ausgereifte Online-Dokumentation auf <https://docs.checkmk.com>.

Egal ob Buch, Online-Dokumentation oder Training: Es lohnt sich, die vielen weiteren Fähigkeiten von Checkmk zu entdecken. Zu ihnen zählen beispielsweise:

### ► Automatische Analyse des Netzwerks

Checkmk kann neue, bislang nicht inventarisierte Hosts in Ihrem Netzwerk erkennen und automatisch anlegen. So vermeiden Sie eine Schatten-IT, vergessene Testsysteme oder den gar nicht mal so seltenen Fall, dass ein scheinbar unwichtiges Testsystem schon lange produktiv gegangen, im Monitoring jedoch nie richtig eingerichtet worden ist.

### ► Überwachung von weiteren Geräten per SNMP

Gerade Blackboxes wie Switches, SAN-Speicher oder Drucker können oft über eine SNMP-Schnittstelle überwacht werden. Aus Platz- und Komplexitätsgründen müssen wir diesen Bereich in diesem Buch leider ausklammern, in der Checkmk-Online-dokumentation finden Sie unter <https://docs.checkmk.com/latest/de/snmp.html> jedoch Weitergehendes.

► **Beliebige weitere Checks durch eigene Plug-ins**

Über das Plug-in-System des auf den Hosts installierten Checkmk-Agenten lassen sich sehr schnell beliebige weitere (Shell-)Skripte hinzufügen, die ganz bestimmte oder vielleicht auch wirklich sehr spezielle Monitoring-Aufgaben übernehmen. Auf diese Art und Weise gibt es quasi fast nichts, was Sie nicht überwachen könnten.

Auch für ungewöhnliche oder selbst geschriebene Software können Sie so beliebige Monitoring-Aufgaben wahrnehmen. Neben der doch recht abstrakten Frage: »Läuft sie oder läuft sie nicht?« gibt es zahlreiche weitere Fragen, die im produktiven Betrieb im Blick behalten werden müssen:

- die aktuelle Aufgabenlast in einer Software
- Füllstände von Queues oder Spool-Verzeichnissen
- die Anzahl der aktuell zeitgleich eingeloggtten Anwender
- bestimmte Fehlerbilder pro Zeiteinheit
- und vieles, vieles mehr

Im Prinzip kann man sagen: Alles, was Sie als Administrator manuell per Shell-Kommandos in Erfahrung bringen, um ein System zu debuggen, könnten Sie auch mithilfe eines Plug-ins im Checkmk-Agenten fortlaufend überwachen lassen. Sie müssen es nur skripten und als Plug-in einbinden.

Die Online-Dokumentation kennt dazu eine eigene Sektion zu »Lokalen Checks«, die Sie unter <https://docs.checkmk.com/latest/de/localchecks.html> finden.

## 29.12 Fazit

Auch wenn dieses Kapitel Ihnen das grundsätzliche technische Werkzeug zum Aufbau einer Monitoring-Lösung an die Hand gegeben hat, steht und fällt der Erfolg Ihrer Umsetzung mit einer sorgfältigen Planung der zu überwachenden Dienste.

Überlegen Sie genau, wann Warnungen und Alarme ausgelöst werden. Denken Sie auch daran, dass nicht jeder kurzzeitige Peak eines Messwertes gleich zum Alarm führen sollte – geben Sie einer Situation (wenn möglich) zunächst ein paar Minuten, um sich zu entspannen. Andernfalls riskieren Sie, von häufigen Fehlalarmen derart genervt zu sein, dass Sie die wirklich wichtigen Ereignisse nicht mehr wahrnehmen. Hier wird Ihnen der Praxiseinsatz im Laufe der Zeit die nötigen Erfahrungswerte vermitteln.

TEIL VII

# **Sicherheit, Verschlüsselung und Zertifikate**



# Kapitel 30

## Sicherheit

*Zusätzlich zu den bereits vorgestellten Maßnahmen zur Steigerung der Sicherheit Ihrer IT-Infrastruktur erfahren Sie in diesem Kapitel, wie Sie Ihre Server und Ihre Infrastruktur weiter absichern können – inklusive chroot, AppArmor, snort (IDS/IPS), fail2ban und OpenVPN.*

In Zeiten von Viren, Würmern, Trojanern und Datenpannen im großen Stil bekommt die Sicherheit von IT-Landschaften eine immer größere Bedeutung. In diesem Buch haben Sie bereits einige Aspekte der Sicherheit kennengelernt, zum Beispiel wie Sie Ihre Netzwerksicherheit erhöhen können (siehe Abschnitt 22.6, »Firewalls mit netfilter und iptables«).

Die oft zitierte Grundlage von Sicherheit in modernen Umgebungen darf trotzdem nicht außer Acht gelassen werden:



*Sicherheit ist ein Konzept, das **vor** dem Bildschirm beginnt!*

Egal welche Anstrengungen Sie unternehmen – wenn Ihre Mitarbeiter keinen verantwortungsvollen Umgang mit der Technik beherrschen, ist der GAU nicht weit entfernt. Das soll jedoch nicht heißen, dass Sie Ihre Umgebung ungeschützt lassen sollen. Ganz im Gegenteil: Mit ein paar kleinen Mitteln können Sie die Sicherheit deutlich verbessern und gegebenenfalls auch ungeübten Nutzern peinliche Fehler und Erklärungen ersparen. Sicherheitstechnik in der EDV war, ist und bleibt ein Hase-und-Igel-Spiel. Dies liegt in der Natur der Sache, da immer zuerst der Virus, die Spam-Mail oder der Exploit analysiert werden muss, um geeignete Gegenmaßnahmen finden zu können. Immer?

Nein, dies ist so nicht korrekt, da durch den Einsatz von IDS/IPS (*Intrusion Detection and Intrusion Prevention System*) auch Angriffe erkannt und unterbunden werden können, die noch nicht über die gängigen Methoden analysiert wurden. Ursächlich für alle Gefahren sind drei Elemente, die sich gegenseitig bedingen oder vor-aussetzen. In Abbildung 30.1 wird dies verdeutlicht. Auch wenn Sie Ihre Anwender intensiv schulen, ist die Gefahr dennoch nicht gebannt. Andersherum können Sie mit unzähligen Firewallkaskaden ebenso wenig eine vollständige Sicherheit Ihres Umfelds generieren.

Es kommt, wie so oft, auf das goldene Mittelmaß an und darauf, dass Sie die Ihnen zur Verfügung stehenden Mittel auch ausschöpfen. Denken Sie aber stets daran, alle Gefahrenfelder im Auge zu behalten, sodass kein Ungleichgewicht entsteht. Was nutzen sechs kaskadierte

Firewalls und ein IDS/IPS-Cluster, wenn Ihre Anwender mit Administratorrechten arbeiten und sich so Schadsoftware leicht einnisten und verbreiten kann.

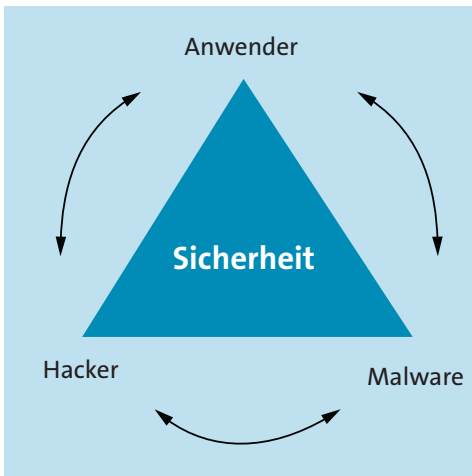


Abbildung 30.1 Die Dreifaltigkeit der Gefahren

## 30.1 Weniger ist mehr

Serverseitig beginnt die Sicherheit bereits bei der Planung. Selbstverständlich sind moderne Systeme in der Lage, Mail-, Spamfilter-, Proxy-, Antiviren-, News-, DNS- und Webserver auf einer Hardware zu betreiben, allerdings erhöht das die Sicherheit nicht. Durch die Vielzahl von Programmen werden große Angriffsflächen geboten, die mögliche Angreifer in die Lage versetzen, nicht nur einen Dienst Ihres Hauses auszuschalten, sondern direkt die gesamte Struktur. Daher empfiehlt es sich, so wenige Dienste wie möglich auf einem System zu betreiben, wobei diese natürlich auf dem neuesten Stand gehalten werden müssen.

Dem geübten Windows-Nutzer und dem an Technik interessierten Zeitungsleser ist natürlich bekannt, dass ein System mit ungepatchten Sicherheitslücken heutzutage nicht mehr lange seinen Dienst verrichtet. Darüber hinaus müssen Sie beachten, dass ein komplexes Umfeld nicht nur fehleranfälliger wird, sondern dass auch leicht Komponenten übersehen werden können, die dann wiederum schnell kompromittiert werden können. Daher sollten Sie sich an den Grundsatz »Keep it simple!« halten.

## 30.2 chroot

Eine einfache Methode, um die Sicherheit zu erhöhen, wird Ihnen in allen Linux-Distributionen bereits frei Haus geliefert – das *chroot*. Ein *chroot* stellt eine Verschiebung des Wurzel-

verzeichnisses für einen Dienst/Prozess dar. Wenn ein Dienst oder Prozess innerhalb eines *chroot* läuft, kann auf Dateien außerhalb des *chroot* nicht zugegriffen werden. Man spricht in diesem Zusammenhang auch von einer sogenannten *Sandbox*<sup>1</sup>. Mittels eines *chroot* können folgende Szenarien erstellt werden:

- ▶ **Honeypot** – Vortäuschung eines echten Systems: Angreifer laufen ins Leere.
- ▶ **Jail** – Benutzer oder Dienste werden in einen abgeschotteten Bereich geleitet, in dem sie nur stark eingeschränkte Möglichkeiten vorfinden.
- ▶ **Reparatur** – Mittels einer Live-CD kann das bestehende System eingebunden werden, um anschließend auf diesem zu arbeiten.

#### Diskussionsstoff: chroot

Trotz dieser Möglichkeiten ist die Klassifizierung des *chroot* als Sicherheitsfeature umstritten. Diese Diskussion basiert unter anderem darauf, dass es trotz des Konzepts von *chroot* Möglichkeiten gibt, die Umgebung zu verlassen, die sogar in der Manpage dokumentiert sind (siehe *chroot(2)*). Dies ist zwar nur als *root* möglich, stellt aber dennoch ein Risiko dar.



### 30.2.1 Dienste

Viele Dienste bieten von Haus aus die Möglichkeit, sie in einer *chroot*-Umgebung zu betreiben. Ein Paradebeispiel liefert uns der bekannte FTP-Dienst. In den meisten auf Sicherheit ausgelegten Variationen steht Ihnen die Möglichkeit offen, den FTP-Daemon im *chroot* zu betreiben, um zum einen den Dienst abzusichern und zum anderen die Benutzer in ihren Homeverzeichnissen einzusperren. Dies ist vor allem dann sinnvoll, wenn nicht alle Daten von allen Benutzern gelesen werden sollen. Wir verwenden den *vsftpd* (*Very Secure FTP Daemon*). Wie der Name schon sagt, ist dieser FTP-Daemon auf Sicherheit ausgelegt. Nach der Installation über die Paketquellen finden wir unter */etc/* oder */etc/vsftpd* die Konfigurationsdatei *vsftpd.conf* (siehe Listing 30.1), in der bereits einige Parameter gesetzt sind.

```
listen=NO
listen_ipv6=YES
anonymous_enable=NO
local_enable=YES
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
```

<sup>1</sup> Eine *Sandbox* (engl. für *Sandkasten*) ist eine geschlossene Umgebung, die nicht verlassen werden kann.

```
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=NO
```

**Listing 30.1** »vsftpd«: Standardkonfiguration nach der Installation (Debian/Ubuntu)

In dieser Grundkonfiguration läuft der Dienst auf allen Distributionen im *Standalone*<sup>2</sup>-Modus auf dem TCP-Port 20. Je nach Distribution sind weitere Konfigurationen vorhanden. Passen Sie folgende Parameter an, um die Sicherheit zu erhöhen:

- ▶ `anonymous_enable=NO`  
keine anonymen Zugriffe erlauben
- ▶ `local_enable=YES`  
erlaubt das Anmelden der lokalen Benutzer.
- ▶ `write_enable=YES`  
erlaubt das Hochladen von Dateien.
- ▶ `ftpd_banner=Willkommen auf meinem sicheren FTP-Server`  
gibt bei der Anmeldung den nachfolgenden Text aus.
- ▶ `chroot_local_user=YES`  
sperrt die lokalen Benutzer in ihre Homeverzeichnisse ein.

Standardmäßig haben Benutzer auf Linux-Systemen an Ihrem eigenen Home-Verzeichnis Schreibrechte, das im Kontext des *vsftp* nun das Root-Verzeichnis darstellt – das mag *chroot* nicht. Daher müssen Sie der *vsftpd*-Konfiguration die Zeile `allow_writeable_chroot=YES` spendieren. Wenn Sie den Home-Verzeichnissen das Schreibrecht entziehen würden, dann könnten die Benutzer dort keine Daten mehr ablegen.



### CentOS und openSUSE: Besonderheiten

Auf CentOS- und openSUSE-Systemen müssen Sie nachstehende Befehle zusätzlich absetzen, damit die lokale Firewall den FTP-Zugriff erlaubt und Benutzer im *chroot* Zugriff auf ihre Homeverzeichnisse erhalten:

```
firewall-cmd --permanent --add-port=21/tcp
firewall-cmd --reload
```

Nach einem `systemctl restart vsftpd.service` können sich alle lokalen Benutzer an Ihrem FTP-Server anmelden. Falls Sie Ihren FTP-Server auf einem Webserver betreiben und Ihren Kunden lediglich einen FTP-Zugriff gewähren wollen, passen Sie die lokale `/etc/passwd` an, um weitere Einschränkungen vorzunehmen. Verlagern Sie das Homeverzeichnis der Benut-

<sup>2</sup> *Standalone*: Der Start des Dienstes erfolgt über ein *systemd*-Skript und nicht über *inet.d*.



zer auf das Verzeichnis des Webangebots, und unterbinden Sie zusätzlich den SSH-Zugang über eine Anpassung der `/etc/passwd`. Ein entsprechend angepasster `passwd`-Eintrag für den Benutzer `tom` sieht so wie in Listing 30.2 aus:

```
tom:x:1005:1005:,,,:/var/www/example.com:/bin/false
```

**Listing 30.2** »passwd«: Einschränkung des Benutzers »tom«

#### **Stolperfalle: /etc/shells**

Durch die Verlinkung der Shell des Benutzers `tom` auf `/bin/false` wird der SSH-Zugriff unterbunden. Der `vsftpd` lässt nur Benutzer mit einer gültigen Shell zu, `/bin/false` ist aber nicht bei allen Distributionen eine gültige Shell! Kontrollieren Sie, ob `/bin/false` unter `/etc/shells` eingetragen ist, und ergänzen Sie diese gegebenenfalls.



### 30.3 Selbstabsicherung: AppArmor

Trotz aller Sicherheitsmaßnahmen bleiben Systeme angreifbar, vor allem bei den Diensten, die sie anbieten. Auch ein ALG<sup>3</sup> oder eine IPS können nicht alle Bereiche abdecken. Hier müssen Sie den Dienst direkt auf dem System schützen. Durch die *Linux Security Module*-Schnittstelle kann Software als Kernelmodul laufen und so direkt die Zugriffsrechte einzelner Prozesse auf höchster Systemebene einschränken. Dahinter steht das MAC<sup>4</sup>-Konzept. Das Projekt *SubDomain* wurde von der Firma Wirex Communications, Inc., später Imunix, gestartet. Im Jahr 2005 wurde es von Novell gekauft, in AppArmor umbenannt und bis 2007 weiterentwickelt. Da es aber den Sprung in den Linux-Kernel immer wieder verfehlte, sollte die Entwicklung eingestellt werden. Die Entwickler um Crispin Cowan wurden entlassen und wollten das Projekt in der neu zu gründenden Firma *Mercenary Linux* weiterführen.

Seit 2009 bemühte sich *Canonical* darum, AppArmor in den Linux-Kernel zu bringen, was ab dem Kernel 2.6.36 gelang, und seither ist es auch per Default bei Ubuntu aktiv (daher verwenden wir als System in diesem Abschnitt auch ein Ubuntu 22.04 LTS).

Das Tool AppArmor arbeitet als Kernelmodul und lädt spezifische Regeln nach, die sogenannten *Profiles*. Neben vordefinierten Regeln können auch eigene hinzugefügt werden. Das Konzept von AppArmor sieht vor, den Betrieb von Software zu analysieren und dazu Regeln zu erstellen, die lediglich die als »normal« spezifizierten Aktionen erlauben. Wird die Software mit Schadcode infiziert, kann dieser die durch AppArmor geschaffene Umgebung nicht verlassen, sich also nicht weiterverbreiten oder unzulässige Aktionen ausführen (zum Beispiel Spam versenden).

<sup>3</sup> *Application Layer Gateway*, Firewall auf Applikationsebene (OSI-Layer 7).

<sup>4</sup> *Mandatory Access Control*, engl. für *zwingend erforderliche Zugangskontrolle*.

Nach der Installation der Pakete *apparmor* und *apparmor-utils* steht Ihnen AppArmor zur Verfügung. Im Regelfall sind diese Pakete auf Ubuntu-Systemen bereits installiert. AppArmor läuft, ohne dass Sie es merken, da die Profile Dienste im Hintergrund schützen. Unter */etc/apparmor.d/* finden Sie die Regelsätze. Die Benennung der Regelsätze enthält den vollständigen Pfad. Allerdings wird der Slash durch einen Punkt ersetzt. Somit heißt das Profil für *tcpdump* entsprechend *usr.sbin.tcpdump*.

### 30.3.1 Status und Betriebsarten

Selbstverständlich bietet AppArmor Ihnen auch Möglichkeiten an, den aktuellen Status und die jeweilige Betriebsart anzeigen zu lassen. Sie rufen den Status von AppArmor über folgenden Befehl ab:

```
root@example:/# apparmor_status
apparmor module is loaded.
30 profiles are loaded.
30 profiles are in enforce mode.
 [...]
 /usr/bin/man
 /usr/lib/NetworkManager/nm-dhcp-client.action
 /usr/lib/NetworkManager/nm-dhcp-helper
 /usr/lib/connman/scripts/dhclient-script
 /usr/sbin/tcpdump
 /usr/sbin/dhclient
 lsb_release
 man_filter
 man_groff
 nvidia_modprobe
 nvidia_modprobe//kmod
 snap-update-ns.lxd
 [...]
0 profiles are in complain mode.
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

**Listing 30.3** »apparmor\_status« auf Ubuntu 22.04

Das Paket *apparmor-utils* stellt Ihnen auch die Möglichkeit zur Verfügung, den Status mit dem Befehl *aa-status* abzurufen. Wie Sie der Ausgabe in Listing 30.3 entnehmen können, sind auf einem aktuellen Ubuntu-22.04-System 30 Profile enthalten, und alle davon sind im *enforce mode* geladen. AppArmor unterscheidet zwei Betriebsarten:

► **enforce mode**

Der »erzwungene Modus« setzt die geladenen Profile um.

► **complain mode**

Der »Beschwerde-Modus« protokolliert lediglich die Verletzungen des Profils.

Alle Meldungen werden sowohl in *syslog* als auch in *kern.log* protokolliert. Hauptsächlich wird aber das *syslog* zur Auswertung benutzt. Über die Befehle `aa-enforce <PROFIL>` und `aa-complain <PROFIL>` bringen Sie das Profil eines Dienstes in den einen oder anderen Modus. Zum vollständigen Beenden und Entladen aller Profile von AppArmor muss das `systemd`-Skript ausgeführt werden und zusätzlich der Befehl `aa-teardown` verwendet werden:

```
root@example:~# systemctl stop apparmor.service
root@example:~# aa-teardown
[...]
```

**Listing 30.4** Beenden und Entladen aller Profile von AppArmor

Dies ist leider notwendig, da nur durch das Stoppen des AppArmor-Dienstes die zurzeit geladenen Profile trotzdem aktiv bleiben. Durch die Installation des Pakets *apparmor-profiles* erhalten Sie viele weitere Profile. Dementsprechend umfangreicher erscheint dann auch die Ausgabe des AppArmor-Status:

```
root@example:~# aa-status
apparmor module is loaded.
48 profiles are loaded.
31 profiles are in enforce mode.
 /usr/bin/man
 /usr/lib/NetworkManager/nm-dhcp-client.action
 /usr/lib/NetworkManager/nm-dhcp-helper
 [...]
17 profiles are in complain mode.
 avahi-daemon
 dnsmasq
 dnsmasq//libvirt_leaseshelper
 [...]
0 profiles are in kill mode.
0 profiles are in unconfined mode.
0 processes have profiles defined.
0 processes are in enforce mode.
0 processes are in complain mode.
0 processes are unconfined but have a profile defined.
0 processes are in mixed mode.
0 processes are in kill mode.
```

**Listing 30.5** »aa-status« mit »apparmor-profiles«

Selbstverständlich ist es möglich, eigene Profile zu erstellen oder vorhandene zu erweitern. Eine Liste der noch nicht von AppArmor geschützten Programme/Dienste erhalten Sie mit dem Tool *aa-unconfined*:

```
root@example:/# aa-unconfined
1 /usr/lib/systemd/systemd (/sbin/init) not confined
622 /usr/lib/systemd/systemd-networkd (/lib/systemd/systemd-networkd) not confined
624 /usr/lib/systemd/systemd-resolved (/lib/systemd/systemd-resolved) not confined
691 /usr/sbin/sshd (sshd: /usr/sbin/sshd -D [listener] 0 of [...]) not confined
2333 /usr/sbin/vsftpd not confined
4142 /usr/sbin/named confined by 'named (enforce)'
3376 /usr/sbin/in.tftpd not confined
```

### Listing 30.6 Ungeschützte Dienste anzeigen

Das Tool ermittelt via *ss* alle Dienste mit offenen TCP- und UDP-Ports und prüft, ob für diese Dienste ein Profil existiert. Wie Sie Listing 30.6 entnehmen können, laufen auf dem System ein *sshd*, ein *vsftpd* und ein *tftpd-hpa* (*in.tftpd*), die noch nicht von AppArmor geschützt werden, sowie ein *named*, der geschützt ist.



#### Eigene Profile

Es sei eindringlich darauf hingewiesen, dass eigene AppArmor-Profilen schnell zu fehlerhaftem Verhalten von Programmen führen können. Im schlimmsten Fall können Sie das Programm, das Sie eigentlich schützen wollen, gar nicht mehr benutzen. Auf der anderen Seite ist es ebenso möglich, dass ein zu »großzügiges« Profil keinen Sicherheitsgewinn mit sich bringt.

### 30.3.2 Eigene Profile erstellen

Wir erstellen nun ein AppArmor-Profil für den *tftpd-hpa*, der uns einen *Trivial File Transfer Protocol*-Daemon zur Verfügung stellt. Listing 30.7 zeigt den notwendigen Aufruf mit dem Tool *aa-genprof*:

```
daniel@server:~$ sudo aa-genprof in.tftpd
Updating AppArmor profiles in /etc/apparmor.d.
Writing updated profile for /usr/sbin/in.tftpd.
Setting /usr/sbin/in.tftpd to complain mode.
[...]
Profiling: /usr/sbin/in.tftpd
```

Please start the application to be profiled in another window and exercise its functionality now.

Once completed, select the "Scan" option below in order to scan the system logs for AppArmor events.  
[...]

```
[(S)can system log for AppArmor events] / (F)inish
```

**Listing 30.7** Eigenes Profil erstellen für den »in.tftpd«

Starten und stoppen Sie den Dienst nun, verbinden Sie sich von einem TFTP-Client mit dem Server, und führen Sie alle gängigen Aktionen aus – hier also lediglich das Verbinden, Herauf- und Herunterladen von Dateien. Anschließend lassen Sie *aa-genprof* durch Eingabe des Buchstabens S einen Scan durchführen. Sie erhalten die Ausgabe aus Listing 30.8:

```
Reading log entries from /var/log/syslog.
Complain-mode changes:
```

```
Profile: /usr/sbin/in.tftpd
Capability: net_bind_service
Severity: 8
```

```
[1 - include <abstractions/nis>]
 2 - capability net_bind_service,
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish
```

**Listing 30.8** »aa-genprof«-Scan für den »in.tftpd«

Nun müssen Sie entscheiden, ob für den *in.tftpd* die angezeigten Profile inkludiert werden sollen oder ob nur der Aufruf von *capability net\_bind\_service* gültig ist. Über die vorangestellten Zahlen können Sie die Auswahl wechseln, was durch die umschließenden eckigen Klammern dargestellt wird. Über das in der letzten Zeile aufgeführte Menü können Sie den Zugriff erlauben, verweigern, beobachten, den Vorgang abrechnen oder beenden. Das Tool führt Sie nun Schritt für Schritt durch alle Bereiche.

Wenn Sie zu den Dateizugriffen gelangen, werden die Auswahlmöglichkeiten umfangreicher (siehe Listing 30.9):

```
Profile: /usr/sbin/in.tftpd
Path: /srv/tftp/download.me
New Mode: rk
Severity: 4
```

```
[1 - /srv/tftp/download.me rk,]
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) /
Abo(r)t / (F)inish
```

**Listing 30.9** »aa-genprof«-Scan: »in.tftpd«-Dateizugriffe

Mittels (G)lob wird ein Zugriff auf das gesamte Verzeichnis zugelassen oder aber lediglich ein Zugriff auf Dateien mit der gleichen Endung: Glob with (E)xtension. Bei der Eingabe von G wird Ihnen jeweils eine tiefere Verzeichnisebene angeboten.

Haben Sie die entsprechende Tiefe erreicht, können Sie den Zugriff erlauben, indem Sie den Buchstaben A eingeben. Listing 30.10 zeigt den Zugriff auf `/srv/tftp/`. Die aktive Auswahl wird stets durch eckige Klammern symbolisiert.

```
Profile: /usr/sbin/in.tftpd
Path: /srv/tftp/download.me
New Mode: rk
Severity: 4
```

```
 1 - /srv/tftp/download.me rk,
 [2 - /srv/tftp/* rk,]
(A)llow / [(D)eny] / (I)gnore / (G)lob / Glob with (E)xtension / (N)ew / Audi(t) /
 Abo(r)t / (F)inish
```

**Listing 30.10** »aa-genprof«-Scan: »in.tftpd«-Verzeichniszugriffe

In Tabelle 30.1 finden Sie eine Übersicht über die Parameter, die AppArmor anwendet.

| Parameter  | Bedeutung                                                                                                        |
|------------|------------------------------------------------------------------------------------------------------------------|
| Profile    | beschreibt das Profil, für das die Regel gilt.                                                                   |
| Execute    | gibt an, welches Programm ausgeführt werden soll.                                                                |
| Path       | legt fest, auf welche(s) Datei/Verzeichnis zugegriffen wird.                                                     |
| Mode       | Zugriffsart: r = Lesen, w = Schreiben, l = folgt einem symbolischen Link.                                        |
| Severity   | Wert der Gefährdung: von 1 = bedenkenlos bis 10 = hohes Risiko                                                   |
| Inherit    | Vererbung des Profils auf <i>child-processes</i> . Nur verwenden, wenn die gleichen Zugriffe vorgenommen werden! |
| Profil     | Programmstart über das Profil. Existiert kein Profil, wird der Start unterbunden.                                |
| Unconfined | normaler Programmstart                                                                                           |

**Tabelle 30.1** »aa-genprof« – Parameter

Folgende Optionen stehen Ihnen für die Parameter zur Verfügung:

- ▶ **Allow**  
erlaubt den Zugriff.

- ▶ **Deny**  
verbietet den Zugriff bzw. den Aufruf der Datei.
- ▶ **Glob**  
erlaubt den Zugriff auf Verzeichnisebene (bei mehrfacher Ausführung).
- ▶ **Glob with (E)xtension**  
wie Glob; mit der Einschränkung auf gleiche Dateieindung
- ▶ **New**  
erstellt eine eigene Regel durch reguläre Ausdrücke.
- ▶ **Abort**  
bricht den Vorgang des Profilerstellens ab.
- ▶ **Finish**  
beendet den Vorgang und erstellt das Profil anhand der bisher erfassten Regeln.

Am Ende der Verarbeitung wird das erstellte Profil durch Eingabe von `S` gespeichert und anschließend mittels `F` verlassen. Das soeben erstellte Profil finden Sie unter `/etc/apparmor.d/usr.sbin.in.tftpd`. Das Profil wird von `aa-genprof` umgehend in den `enforce mode` versetzt und in AppArmor geladen. Wenn es im Regelbetrieb zu Fehlern des Programms kommt, können Sie das erstellte Profil nochmals überarbeiten. Hierfür verwenden Sie den Befehl `aa-logprof`, der eine erneute Prüfung der Logfiles bewirkt und Sie erneut mit der Frage konfrontiert, welche Zugriffe gültig sind.

## 30.4 Gotcha! Intrusion-Detection-Systeme

Auch der ambitionierte Administrator kann nicht immer alles sehen, was in seinem Netzwerk geschieht. Es hilft auch nicht wirklich, ständig einen Sniffer laufen zu lassen – schließlich will der Datenwust auch ausgewertet werden. Dafür gibt es aber automatisierte Tools, die uns die Arbeit abnehmen und die Pakete analysieren und kategorisieren. Das Zauberwort der letzten Jahre heißt *IDS/IPS* (Intrusion Detection System/Intrusion Prevention System). Diese oft als »Allheilmittel« angepriesenen System analysieren den Datenverkehr im Netzwerk und können Angriffe bzw. Schadcode protokollieren, eine Warnmeldung ausgeben (IDS) oder Angriffe sogar unterbinden (IPS).

Diese Systeme arbeiten ähnlich wie Antivirensoftware mit Signaturen, um so Angriffe bzw. Schadcode zu entdecken. Zusätzlich werden noch weiter gehende Analysen vorgenommen, die einem Layer-7-ALG ähneln. Durch den Betrieb im Datenstrom können die Systeme bei entsprechender Konfiguration nicht nur vor Netzwerkangriffen schützen, sondern sogar vorhandene Sicherheitslücken beheben (eigentlich deren Ausnutzung unterbinden), bevor ein Patch des Herstellers veröffentlicht wird. Alle IDS/IPS-Implementationen kämpfen dabei mit zwei großen Hürden:

- ▶ **false positives**  
fälschlich unterbundener Datenverkehr
- ▶ **false negatives**  
gefährlicher Datenverkehr, der nicht erkannt wurde

Die Konfiguration und das nachhaltige Aktualisieren eines IDS/IPS sind von essenzieller Wichtigkeit. Leicht unterbinden Sie durch eine falsche Konfiguration gültigen Datenverkehr oder lassen ungültigen zu. Der Betrieb eines IDS/IPS fordert vor allem andauernde Aufmerksamkeit von Ihnen – das Prinzip des »install and forget« greift hier leider nicht.

Trotz dieser Einschränkungen bzw. möglicher Fehlerquellen kann ein IDS/IPS an den richtigen Stellen nicht nur Ihre Sicherheit erhöhen, sondern Ihnen auch einen tieferen Einblick in Ihr Netzwerk gewähren. In diesem Abschnitt erfahren Sie alles zum Projekt *snort*, von der Installation und Konfiguration bis hin zum wirksamen Betrieb in Ihrer Umgebung.

### 30.4.1 *snort* und Co.

Auf der Projektseite [www.snort.org](http://www.snort.org) findet sich der Satz »Snort has become the de facto standard for IPS«, was gar nicht so weit von der Realität entfernt ist. Das Tool ist ein Paketdeko-der, der den Datenverkehr in Ihrer Umgebung analysiert und somit nach dem Prinzip eines IDS/IPS arbeitet.

Das von Martin Roesch 1998 gestartete Projekt wurde schnell in die von ihm gegründete Firma *Sourcefire* überführt, die *snort* als kommerzielles Produkt unter dem Namen *sourcefire* vertreibt. Diese Firma wurde im Jahre 2014 durch Cisco übernommen. Auch seit der Übernahme wird *snort* in einer Open-Source-Version angeboten, die durch eine kommerzielle Version von Cisco erweitert werden kann, die umfassendere Sicherheitsfunktionen bietet. Entscheidend neben dem Betrieb der Software ist für ein IDS/IPS die Aktualität der Angriffserkennungsmuster. Die Verarbeitung findet bei *snort* in vier Phasen statt:

- ▶ **sniffer**  
Aufnahme der Kommunikation
- ▶ **preprocessor**  
Vorbereitung für die Erkennungseengine und/oder eigene Verarbeitung
- ▶ **detection engine**  
Angriffserkennung durch Signaturen, die aus den *Rulesets* stammen
- ▶ **output-Plugins**  
Aufbereitung der Auswertung und gegebenenfalls Speicherung der relevanten Daten

Bei *snort* sorgen neben den Signaturen die sogenannten *preprocessors* für einen weiteren Schutz. Diese Präprozessoren werten die Rohdaten aus, bereiten diese für etwaige Signaturen vor oder greifen auch direkt in den Datenverkehr ein.



Den neuesten Zuwachs stellt *openappID* dar. Damit kann *snort* Applikationen erkennen und wird somit zu einer Next-Generation-Firewall erweitert. *snort* kann als reines IDS an einem beliebigen *Mirrorport*<sup>5</sup> Ihres Netzwerks betrieben werden und Sie vor gegebenenfalls stattfindenden Angriffen oder übertragenem Schadcode warnen. Darüber hinaus können Sie *snort* auch als *Inline*<sup>6</sup>-IPS verwenden, sodass Attacken oder Schadcode von *snort* direkt unterbunden werden kann. Durch die Unterteilung der Software in ein freies und ein kommerzielles Angebot werden auch entsprechend unterschiedliche Updates zur Verfügung gestellt. Sie können zwischen folgenden Szenarien wählen:

- ▶ **Community**  
inoffizielle Sammlung von Rules der Community
- ▶ **Statische Rulesets**  
Major-Release-Updates, die nur mit neuen *snort*-Versionen ausgeliefert werden
- ▶ **Registered User Release**  
Nach einer Registrierung erhält man Updates, die 30 Tage älter sind als die des kostenpflichtigen Modells.
- ▶ **Subscription Release**  
kostenpflichtig, dafür aber topaktuell

Bei den ersten beiden Punkten handelt es sich um Offline-Angebote, die Sie herunterladen und installieren können. Die anderen beiden werden online zur Verfügung gestellt, sodass aktuelle Versionen Ihrer Installation hinzugefügt werden. Ein IDS/IPS lebt wie eine Antivirensoftware von der Aktualität. Daher empfehlen wir Ihnen dringend, ein Online-Angebot einzusetzen. Die Erkennungsrate Ihres Systems wird deutlich steigen.



## 30.5 Installation und Konfiguration

Die aktuelle Version 3.1.47.0 erhalten Sie als Quellcode auf der Projektseite [www.snort.org](http://www.snort.org). Bei Ubuntu und Debian ist *snort* Bestandteil der Paketquellen – allerdings in einer älteren Version (2.9.15.1). Analog zu Virenscannern sollte *snort* stets in der aktuellsten Version betrieben werden. Daher zeigen wir Ihnen, wie Sie *snort* und alle benötigten Komponenten selbst kompilieren können. Folgende Komponenten werden wir einrichten:

- ▶ **daq**: Abstraktionsebene zwischen *snort* und *PCAP*
- ▶ **snort**: das eigentliche IDS/IPS
- ▶ **pullpork**: Herunterladen und Aktualisieren der Regelsätze (*Rulesets*)
- ▶ **systemd**: Anlegen der systemd-Skripte zum Starten, Stoppen und Neuladen

5 *Mirrorport*, engl. für *Spiegelschnittstelle*: eine Spiegelung eines Datenports im Netzwerk, der sich meist an einem Netzübergabepunkt befindet.

6 *Inline* ist eine Netzwerkkomponente, die sich im Übertragungsweg befindet.

### 30.5.1 Vorbereitungen

In diesem Abschnitt bereiten wir die Installation vor. Dafür werden die Abhängigkeiten installiert und die Quellen heruntergeladen.

Zunächst legen wir aber einen Benutzer und eine Gruppe für Snort an, unter denen der Dienst später laufen wird:

```
groupadd snort
useradd snort -r -s /sbin/nologin -c "SNORT user" -g snort
```

**Listing 30.11** Benutzer und Gruppe für »snort« anlegen

#### Debian/Ubuntu

Installieren Sie die benötigten Abhängigkeiten so, wie in Listing 30.12 dargestellt:

```
apt install build-essential git libpcap libpcap-dev libpcre3-dev libdumbnet-dev \
 ethtool openssl zlib1g-dev liblzma-dev liblua5.1-dev libssl-dev libnet1-dev \
 libnetfilter-queue-dev libwww-perl libcrypt-ssleay-perl libnghttp2-dev pkg-config \
 autotools-dev libhwloc-dev cmake libunwind-dev libgoogle-perftools-dev
```

**Listing 30.12** Abhängigkeiten installieren (Debian/Ubuntu)

#### openSUSE Leap

Neben der Installation der Abhängigkeiten müssen Sie auch noch zwei Pfadanpassungen vornehmen (symbolische Links). Setzen Sie dafür die Befehle aus Listing 30.13 ab:

```
zypper install -t pattern devel_C_C++ devel_basis
zypper install zlib-devel libpcap libpcap-devel pcre pcre-devel libnet1 \
 libdnet-devel tcpdump cmake libnetfilter_queue-devel libnetfilter_acct-devel \
 texlive-texlua5.1-devel libopenssl-devel perl-Crypt-SSLeay gperftools-devel \
 lua5.1 lua5.1-devel hwloc-devel
```

**Listing 30.13** Abhängigkeiten installieren und Pfadkorrekturen vornehmen

#### CentOS

Unter CentOS können Sie direkt die RPM-Pakete so installieren, wie sie in Listing 30.14 dargestellt werden. Für einige Abhängigkeiten müssen Sie vorher das *epel*-Repository hinzufügen:

```
dnf install -y epel-release
dnf -y groupinstall Development\ Tools
dnf install cmake pcre-devel libpcap libpcap-devel libdnet-devel hwloc-devel \
 lua5.1-devel libunwind-devel gperftools-devel perl-LWP-UserAgent-Determined \
 perl-LWP-Protocol-https perl-Sys-Syslog perl-Archive-Tar
ldconfig
```

**Listing 30.14** Installieren auf CentOS

### 30.5.2 Kompilieren und installieren

Laden Sie die Pakete *libdaq* und *snort* als root-Benutzer herunter, entpacken Sie die Pakete, und installieren Sie diese mit dem üblichen Dreisprung. Allerdings müssen Sie für *snort* je nach Distribution hier und da ein paar Haken schlagen. Hier sehen Sie zunächst die Kommandos für *libdaq*:

```
cd /opt && git clone https://github.com/snort3/libdaq.git
[...]
cd libdaq/
./bootstrap
[...]
./configure
[...]
Build AFPacket DAQ module.. : yes
Build BPF DAQ module..... : yes
Build Divert DAQ module.... : no
Build Dump DAQ module..... : yes
Build FST DAQ module..... : yes
Build netmap DAQ module.... : no
Build NFQ DAQ module..... : no
Build PCAP DAQ module..... : yes
Build Savefile DAQ module.. : yes
Build Trace DAQ module.... : yes
Build GWLB DAQ module..... : yes

make
make install
ldconfig
CentOS: ln -s /usr/local/lib/libdaq.so.3 /lib/
```

**Listing 30.15** Installieren von »libdaq«

Und nun ist *snort* an der Reihe:

```
$ cd /opt
$ git clone https://github.com/snort3/snort3.git
[...]
$ cd snort3
$./configure_cmake.sh --prefix=/usr/local
[...]

snort version 3.1.49.0
[...]
-- Configuring done
```

```
-- Generating done
-- Build files have been written to: /opt/snort3/build
```

```
$ cd build/
$ make && sudo make install
[...]
```

**Listing 30.16** Installieren von »snort«

```
/usr/local/bin/snort --version

,,_ -*> Snort++ <*-
o")~ Version 3.1.49.0
' '' By Martin Roesch & The Snort Team
 http://snort.org/contact#team
 http://snort.org/contact#team
 Copyright (C) 2014-2022 Cisco and/or its affiliates. All rights reserved.
 Copyright (C) 1998-2013 Sourcefire, Inc., et al.
 [...]
```

**Listing 30.17** Prüfen, ob die Installation erfolgreich war

Nach der Installation müssen noch ein paar Anpassungen vorgenommen werden, damit Ihr *snort* betriebsbereit wird. Zunächst müssen Sie einige Verzeichnisse und Dateien erstellen und deren Rechte anpassen:

```
Verzeichnisse
mkdir -p /usr/local/etc/snort/{builtin_rules,rules,appid,intel,lists}
mkdir /var/log/snort /usr/local/etc/snort/rules/iplists/

Rechte
chmod -R 5775 /var/log/snort
chown -R snort:snort /var/log/snort
```

**Listing 30.18** Verzeichnisse/Dateien erzeugen und Rechte anpassen

### 30.5.3 Basiskonfiguration

Noch ist Ihre Installation nicht lauffähig, da noch (mindestens) eine Angabe fehlt. Öffnen Sie dafür die zentrale Konfigurationsdatei */usr/local/etc/snort/snort.lua* mit root-Rechten, und passen Sie den in Listing 30.19 dargestellten Parameter an:

```
Zeile 24:
HOME_NET <NET>
```

**Listing 30.19** Anpassungen in »/usr/local/etc/snort/snort.lua«

Ersetzen Sie den Platzhalter <NET> durch Ihr lokales Netzwerk, also zum Beispiel durch das Netzwerk 192.168.0.0/24. Um unser Arbeitsergebnis zu überprüfen, können wir `snort` mit dem Schalter `-T` im Testmodus starten:

```
$ snort -T -i eth0 -c /usr/local/etc/snort/snort.lua

o")~ Snort++ 3.1.49.0

Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
ssh
host_cache
 [...]

search engine

pcap DAQ configured to passive.

Snort successfully validated the configuration (with 0 warnings).
o")~ Snort exiting
```

**Listing 30.20** Überprüfen der Konfiguration

#### CentOS: Extrabehandlung

Leider benötigt CentOS hier noch eine Pfadangabe. Ergänzen Sie daher zu dem Befehl aus Listing 30.20 noch `--daq-dir /usr/local/lib/daq`, ansonsten wird der Befehl mit Fehlern beendet.

Endet die Ausgabe so, wie in Listing 30.20 dargestellt, ist Ihre Installation startbereit. Wie Sie in Listing 30.20 sehen, wurde zusätzlich mit dem Schalter `-i` die Netzwerkschnittstelle angegeben und mit `-c` der Pfad zur Konfigurationsdatei.

### 30.5.4 Ein erster Test: ICMP

Um zu prüfen, ob `snort` auch wirklich arbeitet, richten wir eine einfache Regel ein, die sämtlichen ICMP-Datenverkehr erfasst. Fügen Sie dafür die Zeilen aus Listing 30.21 zur Datei `/usr/local/etc/snort/rules/local.rules` hinzu:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP detected"; sid:10000001; \
rev:001; classtype:icmp-event;)
```

**Listing 30.21** Regel für »ICMP« in »local.rules«

Bitte beachten Sie, dass der Inhalt von Listing 30.21 in einer Zeile stehen muss. Die Regel besagt, dass ein Alarm (alert) für jeglichen ICMP-Datenverkehr (icmp any any) ausgegeben werden soll, wenn dieser vom lokalen Netzwerk (\$HOME\_NET) stammt. In Klammern werden anschließend die Nachricht und die Klassifizierung festgelegt. Starten Sie nun *snort* mit dem Parameter aus Listing 30.22, um die Ausgabe auf der Konsole zu erhalten. Führen Sie dann einen Ping von einem anderen System auf Ihren Server aus, erhalten Sie diese Ausgabe:

```
$ snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/snort/rules/local.rules \
-i eth0 -A alert_fast -s 65535 -k none
[...]

pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth0
01/07-11:27:15.565463 [**] [1:10000001:1] "ICMP detected" [**] [Classification: \
Generic ICMP event] [Priority: 3] {ICMP} 192.168.0.166 -> 192.168.0.163
01/07-11:27:15.565507 [**] [1:10000001:1] "ICMP detected" [**] [Classification: \
Generic ICMP event] [Priority: 3] {ICMP} 192.168.0.163 -> 192.168.0.166
```

**Listing 30.22** Starten von »snort« mit Ausgabe auf der Konsole

Wie Sie in Listing 30.22 sehen, wurde ein Ping auf das *snort*-System abgesetzt. Dadurch wurden zwei Log-Einträge erzeugt: einer für die Anfrage und einer für die Antwort. *snort* arbeitet also wie erwartet.

### 30.5.5 Start-Skript erstellen: *systemd*

Selbstverständlich wollen Sie *snort* nicht von Hand starten, sondern den Start wie gewohnt von *systemd* erledigen lassen. Dafür erstellen Sie einfach die Datei */etc/systemd/system/snort.service* mit dem Inhalt aus Listing 30.23. Anschließend können Sie *snort* wie gewohnt mit *systemd* bedienen.

```
[Unit]
Description=Snort3 Daemon
After=syslog.target network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/snort -c /usr/local/etc/snort/snort.lua -i <NET> \
-l /var/log/snort -D -u snort -g snort --create-pidfile -k none
ExecReload=/bin/kill -SIGHUP $MAINPID
User=snort
Group=snort
Restart=on-failure
```

```
RestartSec=5s
CapabilityBoundingSet=CAP_NET_ADMIN CAP_NET_RAW CAP_IPC_LOCK
AmbientCapabilities=CAP_NET_ADMIN CAP_NET_RAW CAP_IPC_LOCK
```

```
[Install]
WantedBy=multi-user.target
```

**Listing 30.23** systemd-Skript: »snort.service«

Bitte passen Sie den Platzhalter <NET> an die richtige Netzwerkschnittstelle Ihres Systems an. Lassen Sie abschließend noch die Systemd-Start-Skripte einmal mit `systemctl daemon-reload` neu einlesen. (CentOS: Ergänzen Sie beim Kommando im Parameter `ExecStart` aus Listing 30.23 wieder die Pfadangabe `--daq-dir /usr/local/lib/daq`.)

## 30.6 Immer das Neueste vom Neuen: pulledpork

30

Falls Sie früher bereits *snort* eingesetzt haben, ist Ihnen sicherlich noch der *oinkmaster* ein Begriff. Mit diesem Tool wurden die Regelsätze heruntergeladen und für *snort* bereitgestellt. Dieses Tool wurde durch *pulledpork*<sup>7</sup> ersetzt, das aber sehr ähnlich arbeitet.

Es wird eine Registrierung bei *snort* vorausgesetzt, da für das Herunterladen ein sogenannter *Oinkcode* benötigt wird. Diesen und die dafür notwendige Registrierung können Sie unter [www.snort.org/users/sign\\_up](http://www.snort.org/users/sign_up) einrichten. Sie erhalten umgehend eine E-Mail, in der Sie einen Link zur Bestätigung Ihres Zugangs finden. Nach erfolgreichem Login auf der *snort*-Webseite können Sie unter den Benutzereinstellungen den Punkt `OINKCODE` wählen und diesen Code dort erzeugen. Das Programm selbst wird über GitHub bereitgestellt. Da es sich dabei lediglich um ein Perl-Skript handelt, ist die Installation schnell erledigt (siehe Listing 30.24):

```
Download
cd /opt && git clone https://github.com/shirkdog/pulledpork3.git

Konfiguration anlegen
cd pulledpork3
mkdir /usr/local/etc/pulledpork/
cp etc/pulledpork.conf /usr/local/etc/pulledpork/

Installieren
mkdir /usr/local/bin/pulledpork/
cp pulledpork.py /usr/local/bin/pulledpork/
cp -r lib/ /usr/local/bin/pulledpork/
ln -s /usr/local/bin/pulledpork/pulledpork.py /usr/local/bin/pulledpork.py
```

**Listing 30.24** Installation von »pulledpork«

<sup>7</sup> *pulled pork*, engl. für Zupfbraten.

Anschließend muss die Konfigurationsdatei noch angepasst werden. Vorher ersetzen wir noch den Oinkcode mit `sed` und legen ein paar Verzeichnisse an, die später benötigt werden:

```
$ sed 's#xxxxx#<OINKCODE>#g' -i /usr/local/etc/pulledporkpulledpork.conf
$ mkdir /usr/local/etc/rules/ /usr/local/etc/so_rules/ /usr/local/etc/lists/
```

**Listing 30.25** Korrigieren des Oinkcodes in »pulledpork.conf« und Anlegen von Pfaden

Ersetzen Sie den Platzhalter `<OINKCODE>` (in Fettschrift) durch den zuvor von Ihnen generierten Code. Nun müssen Sie noch ein paar Parameter in der Konfigurationsdatei anpassen. Die Zeilen in Listing 30.26 stellen die Minimalkonfiguration dar:

```
Which Snort/Talos rulesets do you want to download (recomended: choose only one)
community_ruleset = false
registered_ruleset = false
LightSPD_ruleset = true ### Default: false

Your Snort oinkcode is required for snort/talos Subscription, Light_SPD, [...]
oinkcode = <OINKCODE> ### Default: xxxxx

which blocklists to download
snort_blocklist = true ### Default: false
et_blocklist = false
[...]

Where is the Snort Executable located (if not on the system path)
snort_path = /usr/local/bin/snort ### Default: auskommentiert
[...]

What Distro are you running? This must match one of the supported distros:
centos-x64, debian-x64, fc-x64, opensuse-x64, ubuntu-x64
disable this entry to compile rules manually (not yet supported)
distro = <DISTR0> ### Default: ubuntu-x64
[...]
```

**Listing 30.26** Minimalkonfiguration in der »pulledpork.conf«

Im ersten Block müssen Sie sich für einen Regelsatz entscheiden. Dabei haben Sie die Wahl zwischen den folgenden drei Regelsätzen:

- ▶ `community_ruleset`: Standardregeln der Community.
- ▶ `registered_ruleset`: Regeln für registrierte Benutzer – werden aus den kommerziellen Regeln nach 30 Tagen überführt.
- ▶ `LightSPD_ruleset`: Regeln für registrierte Benutzer in einem neuen und schnelleren Format – werden ebenfalls aus den kommerziellen Regeln nach 30 Tagen überführt.



Wir empfehlen Ihnen, den Parameter `LightSPD_ruleset` zu aktivieren, also ihn auf `true` zu setzen. Da alle Regelsätze Überschneidungen haben, ist es nicht ratsam, mehr als einen Regelsatz gleichzeitig zu aktivieren. Anschließend wird der `oinkcode` gesetzt – dank des zuvor abgesetzten `sed`-Kommandos sollte er schon korrekt hinterlegt sein. Im dritten Block müssen Sie sich erneut entscheiden, dieses Mal für eine Blockliste:

- ▶ `snort_blocklist`: Standardblockliste, die von *snort* selbst herausgegeben wird
- ▶ `et_blocklist`: steht für die *Emerging Threats*-Blockliste, deren Inhalt von *Spamhaus*, *DS-hield* und *Abuse.ch* zusammengetragen wurde die und von der US-amerikanischen Firma *Proofpoint* bereitgestellt wird.

Wir empfehlen Ihnen, zunächst mit der `snort_blocklist` zu beginnen. Den vierten Block müssen Sie einkommentieren, damit *pullpork* auch unseren selbst kompilierten *snort* findet. Zuguter Letzt müssen Sie noch den Parameter `distro` an Ihre Distribution anpassen. Die Distributionen sind dabei nicht mit ihren Versionen aufgelistet, sondern lediglich als übergeordnete Angabe. Die Zuordnung kann so erfolgen:

- ▶ Debian 11: `debian-x64`
- ▶ Ubuntu 22.04: `ubuntu-x64`
- ▶ CentOS 9 Stream: `centos-x64`
- ▶ openSUSE Leap: `opensuse-x64`

Zur Kontrolle der Konfiguration können Sie *pulledpork* einfach auf der Konsole ausführen. Rufen Sie das Skript dafür einfach mit dem Schalter `-c`, gefolgt vom Pfad zur Konfigurationsdatei und dem Schalter `-l` auf, um eine Log-Ausgabe zu erhalten. Die Ausgabe sollte so aussehen, wie in Listing 30.27 dargestellt:

```
root@ubuntu:~# pulledpork.py -c /usr/local/etc/pulledpork/pulledpork.conf
```

```
https://github.com/shirkdog/pulledpork3
```

```

 \-----,_____) PuledPork v3.0.0.4 - Lowcountry yellow mustard bbq sauce
 \---==\\ / is the best bbq sauce. Fight me.
 \---==\\ /
 .-~~~~-.Y|__ Copyright (C) 2021 Noah Dietrich, Colin Grady, Michael Shirk
@/_ / / 66__ and the PuledPork Team!
 | \ \ \ _ ("
 \ /-| ||'--' Rules give me wings!
 _ \ _ \
~~~~~

```

```
Loading configuration file: /usr/local/etc/pulledpork/pulledpork.conf
```

```
Processing LightSPD ruleset
```

```
Preparing to modify rules by sid file
```

```
Completed processing all rulesets and local rules:
- Collected Rules: Rules(loaded:48381, enabled:9530, disabled:38851)
- Collected Policies:
  - Policy(name:connectivity, rules:559)
  [...]
Writing rules to: /usr/local/etc/rules/pulledpork.rules
Writing blocklist file to: /usr/local/etc/lists/default.blocklist
Program execution complete.
```

**Listing 30.27** Herunterladen der Regelsätze mit »pulledpork.py«

Damit neue Regelsätze automatisch geladen werden, sollten Sie einen Cronjob anlegen, der dies für Sie übernimmt. Speichern Sie dafür einfach die Datei *pulledpork* mit dem Inhalt aus Listing 30.28 im Verzeichnis */etc/cron.hourly* ab. Damit wird stündlich nach neuen Regelsätzen Ausschau gehalten:

```
#!/bin/bash
/usr/local/bin/pulledpork.py -c /usr/local/etc/pulledpork/pulledpork.conf
```

**Listing 30.28** Einrichten eines Cronjobs für stündliche Updates

## 30.7 Klein, aber oho: fail2ban

Das Tool *fail2ban* analysiert Log-Dateien und blockt IP-Adressen, die zu oft abgewiesen wurden, mittels *iptables/TCP Wrapper*. Das 2004 von Cyril Jaquier ins Leben gerufene Tool steht unter der *GNU General Public License Version 2* frei zur Verfügung.

### 30.7.1 Konfiguration

Nach der Installation aus den Paketquellen finden Sie unter */etc/fail2ban* die Konfigurationsdateien.



#### Besonderheit: CentOS

Bei CentOS ist *fail2ban* standardmäßig nicht enthalten. Sie müssen zunächst die EPEL-Repositories hinzufügen, damit die Installation erfolgen kann:

```
$ dnf install -y epel-release
```

Das Konzept von *fail2ban* ist modular. Die Konfiguration des Dienstes findet in der *fail2ban.conf* statt, die Konfiguration der zu schützenden Dienste in der *jail.conf*, die Filter werden im Verzeichnis */etc/fail2ban/filter.d* abgelegt und die eigentlichen Aktionen im Verzeichnis */etc/fail2ban/action.d*. Durch den modularen Aufbau können Sie ein und denselben Filter für mehrere Dienste verwenden.

In der *fail2ban.conf* finden Sie die globale Konfiguration. Passen Sie folgende Parameter an:

- ▶ `loglevel`  
Log-Verhalten von *fail2ban*
- ▶ `logtarget`  
Speicherort des Logs
- ▶ `socket`  
Ort der Socket-Datei

In der *jail.conf* werden sowohl globale Konfigurationen als auch die Konfiguration der einzelnen »Gefängnisse« für die jeweiligen Dienste vorgenommen. Diese Konfigurationsdatei ist in Sektionen unterteilt. In der Sektion [DEFAULT] wird das generelle Verhalten von *fail2ban* konfiguriert. Einige dort angegebene Parameter können durch das erneute Angeben in einer Sektion überschrieben werden. In den ersten Zeilen dieser Datei steht folgender Hinweis:

```
# Provide customizations in a jail.local file or a jail.d/customisation.local.
# For example to change the default bantime for all jails and to enable the
# ssh-iptables jail the following (uncommented) would appear in the .local file.
```

#### Listing 30.29 Modifizierungshinweis

Dies beachten Sie natürlich und kopieren die vorhandene *jail.conf* wie folgt:

```
root@example:/etc/fail2ban# cp jail.conf jail.local
```

#### Listing 30.30 Kopie: »jail.local«

Wo Log-Dateien zu finden sind, wird über separate Konfigurationsdateien spezifiziert. Diese Konfigurationsdateien heißen stets *path-<SYSTEM>.conf*. Darüber findet *fail2ban* die richtigen Dateien. Aber was ist mit *systemd* und somit mit *journald*? Keine Sorge, ob *fail2ban* sich die Daten aus einer Datei holt oder über *journald*, ermittelt es selbst. Es verwendet standardmäßig die Einstellung `backend = auto`, was *fail2ban* veranlasst, zunächst nach Dateien Ausschau zu halten und dann, wenn es keine passenden findet, *journald* zu verwenden. Passen Sie folgende Parameter Ihren Bedürfnissen entsprechend in der neu angelegten Datei *jail.local* an. Sie finden die Parameter in der Sektion [DEFAULT]:

- ▶ `ignoreip`  
Auf die hier durch Leerzeichen getrennten IP-Adressen, Netzwerke oder DNS-Namen werden keine Regeln angewandt. Konfigurieren Sie hier Ihr lokales Netz oder zumindest Ihre Arbeitsstation, damit Sie im Fehlerfall noch Zugriff auf das System erhalten.
- ▶ `bantime`  
Zeitangabe in Sekunden, wie lange eine IP-Adresse gesperrt wird. Mit der Angabe von -1 wird die IP-Adresse permanent gesperrt.
- ▶ `maxretry`  
Anzahl der maximalen Fehlversuche

- ▶ `destemail`  
Hier legen Sie fest, an welche E-Mail-Adresse Sperrungen gemeldet werden sollen.
- ▶ `banaction`  
Definition des Sperrverfahrens (Default: *iptables-multiport*). Weitere Möglichkeiten: *iptables*, *iptables-new*, *iptables-multiport*, *shorewall* etc.
- ▶ `mta`  
gibt an, welcher MTA zum Mail-Versand genutzt werden soll (Default: *sendmail*).
- ▶ `action`  
gibt an, welche Standardaktion ausgeführt werden soll, wenn ein Filter zutrifft (Default: *nur sperren*). Weitere Möglichkeiten: *Sperren und Mail versenden* oder *Sperren und Mail mit Log-Auszügen versenden*.
- ▶ `findtime`  
Gibt an, innerhalb welcher Zeitspanne die Fehlversuche auftreten müssen, um zu einer Sperrung zu führen. Dieser Parameter ist in der Standardkonfiguration auf 10m und somit auf zehn Minuten eingestellt.

In den Sektionen der Dienste müssen Sie weitere Angaben festlegen. Im folgenden Beispiel erläutern wir die Sektionen `[sshd]` und `[postfix]`:

```
[sshd]
port    = ssh
logpath = %(sshd_log)s
backend = %(sshd_backend)s
[...]
```

```
[postfix]
mode    = more
port    = smtp,465,submission
logpath = %(postfix_log)s
backend = %(postfix_backend)s
```

**Listing 30.31** »jail.local«: Sektion »sshd« und »postfix«

Wie Sie sehen, ist die Konfiguration sehr übersichtlich. Das liegt am modularen Aufbau, denn nur, wenn die Standardwerte verändert werden sollen, müssen diese aktiviert werden. Mit der Direktive `port` wird festgelegt, auf welche Kommunikation *fail2ban* achten soll (im Beispiel für SSH also auf Port 22 und für Postfix auf Port 25, 465 und 587).

Mit `logpath` teilen Sie *fail2ban* mit, wo das jeweilige Log des Dienstes zu finden ist. Wie Sie sehen, sind hier keine Pfade angegeben, sondern Platzhalter. Diese werden über die bereits beschriebenen Pfaddateien aufgelöst. Was beiden Diensten fehlt, ist, dass diese nicht aktiviert sind. Per Default ist *fail2ban* inaktiv.

Im Verzeichnis `/etc/fail2ban/filter.d` liegen die jeweiligen Filterkonfigurationen. Filter werden über reguläre Ausdrücke spezifiziert. Für Postfix sieht der Filter (gekürzt) so aus:

```
[INCLUDES]
before = common.conf

[Definition]
_daemon = postfix(-\w+)?/\w+(?:/smtp[ds])?
_port = (?:\d+)?
prefregex = ^(__prefix_line)s<mdpr-<mode>> <F-CONTENT>.+</F-CONTENT>$
[...]
failregex = <mdre-<mode>>
[...]
ignoreregex =

[Init]
journalmatch = _SYSTEMD_UNIT=postfix.service
```

**Listing 30.32** Auszug des Filters »`/etc/fail2ban/filter.d/postfix.conf`«

Nach dem Inkudieren der Standards (`common.conf`) folgt bereits die Definition des Filters. Die Direktiven `_daemon` und `_port` geben an, auf welche Namen und Ports dieser Filter reagieren soll. Über die Direktive `failregex` werden reguläre Ausdrücke kombiniert, die über den `prefregex`, `mdpr-<MODE>` und `mdre-<MODE>` definiert werden. Über die Konfiguration in der `jail.local` wird der Modus und somit der Regex festgelegt – standardmäßig ist der Modus auf `normal` gesetzt. Der Parameter `ignoreregex` gibt Ausdrücke an, auf die `fail2ban` nicht reagieren soll. Anschließend wird im `Init`-Block festgelegt, wie `fail2ban` an das Journal kommt.

Wollen Sie nun den SSH-Dienst Ihres Systems von `fail2ban` schützen lassen, genügt es, die Datei `sshd.local` mit dem Inhalt aus Listing 30.33 in `/etc/fail2ban/jail.d` zu erstellen:

```
[sshd]
enabled = true
maxretry = 3
bantime = 300
```

**Listing 30.33** »fail2ban«-Überwachung für SSH aktivieren in »`sshd.local`«

In Listing 30.33 wurde nicht nur der Filter `sshd` mit der Direktive `enable` aktiviert, sondern auch die maximale Anzahl von Fehlversuchen mit `maxretry` auf 3 gesetzt (Default: 5) und die Sperrzeit auf 300 Sekunden mit `bantime` verkürzt (Default: 10 Minuten). Starten Sie `fail2ban` nun mit `systemctl restart fail2ban`, wird der SSH-Dienst überwacht. In der Standardkonfiguration legt `fail2ban` für jeden Treffer eines Filters eine `iptables`-Regel in einer eigenen `fail2ban-<DIENSTNAME>`-Chain an – allerdings erst, sobald das erste System gefangen genommen wurde.

### 30.7.2 Aktive Sperrungen

Die aktuell aktivierten Regeln können Sie mit dem Programm `iptables` anzeigen lassen:

```
root@example:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
f2b-sshd   tcp  --  0.0.0.0/0             0.0.0.0/0           multiport dports 22
[...]

Chain f2b-sshd (1 references)
target     prot opt source                destination
REJECT     all  --  192.168.0.164        0.0.0.0/0           reject-with icmp-port-unreachable
RETURN     all  --  0.0.0.0/0           0.0.0.0/0
```

**Listing 30.34** »iptables«-Regeln mit »f2b-sshd« anzeigen lassen

Aus Listing 30.34 geht hervor, dass der Client `192.168.0.164` bereits gesperrt wurde. Dies finden wir in der Log-Datei `/var/log/fail2ban.log` wieder:

```
[...]
2021-01-07 10:17:00,732 fail2ban.jail [3014]: INFO Jail 'sshd' started
[...]
2021-01-07 10:18:27,451 fail2ban.filter [3014]: INFO [sshd] Found 192.168.0.164
2021-01-07 10:18:27,515 fail2ban.actions [3014]: NOTICE [sshd] Ban 192.168.0.164
###
2023-01-07 15:59:03,354 fail2ban.jail [2411]: INFO Jail 'sshd' started
[...]
2023-01-07 16:01:15,210 fail2ban.filter [2411]: INFO [sshd] Found 192.168.0.164
2023-01-07 16:01:15,554 fail2ban.actions [2411]: NOTICE [sshd] Ban 192.168.0.164
```

**Listing 30.35** Log-Auszug in »fail2ban.log«

Nach einem Neustart des `fail2ban` werden alle erstellten `iptables`-Regeln entfernt, sodass Ihr System wieder alle Zugriffe annimmt. Aber selbstverständlich müssen Sie nicht immer die Holzhammermethode verwenden. Mit dem Programm `fail2ban-client` können Sie den aktuellen Status so abfragen, wie es in Listing 30.36 gezeigt wird:

```
root@example:/# fail2ban-client status
Status
|- Number of jail:      1
`- Jail list:  sshd

root@example:/# fail2ban-client status sshd
Status for the jail: sshd
|- Filter
```

```

| |- Currently failed: 1
| |- Total failed:    4
| `-- File list:      /var/log/auth.log
`-- Actions
    |- Currently banned: 0
    |- Total banned:    1
    `-- Banned IP list: 192.168.0.164

```

**Listing 30.36** Statusübersicht mit »fail2ban-client«

Wie Sie in Listing 30.36 sehen, erhalten Sie alle relevanten Informationen übersichtlich aufbereitet. Falls Ihnen *fail2ban* mal zu streng war, können Sie gesperrte IP-Adressen auch wieder freigeben, bevor die Sperrzeit abgelaufen ist. Setzen Sie dafür, um zum Beispiel die gesperrte IP-Adresse 192.168.0.164 wieder freizugeben, den folgenden Befehl ab:

```

root@example:/# fail2ban-client set sshd unbanip 192.168.0.164
1

```

**Listing 30.37** Entsperren einer IP-Adresse mit »fail2ban-client«

### 30.7.3 Reguläre Ausdrücke

Selbstverständlich können Sie *fail2ban* auch um eigene Dienste und Filter ergänzen oder erweitern. Da die Erstellung von regulären Ausdrücken durchaus komplex ist, stellt uns *fail2ban* ein einfaches Kontrollwerkzeug zur Verfügung. Prüfen Sie Ihre regulären Ausdrücke (*Regex*) mit *fail2ban-regex*. Eine Geschwindigkeitsauswertung wird ebenfalls angezeigt:

```

root@example:/# fail2ban-regex '[Thu Jan 07 11:28:53 2023] [error] \
[client 1.1.1.1] user daniel: authentication failure for "/private": \
Password Mismatch' '[[client <HOST>[]] user .* Password Mismatch'

Running tests
=====
Use failregex line : [[client <HOST>[]] user .* Password Mismatch
Use single line : [Sat Sep 20 16:29:03 2018] [error] \ [client 1.1.1...

```

Results

```

=====
Failregex: 1 total
|- #) [# of hits] regular expression
| 1) [1] [[client <HOST>[]] user .* Password Mismatch
`-

```

Ignoreregex: 0 total

```
Date template hits:
|- [# of hits] date format
| [1] {^LN-BEG}(?:DAY )?MON Day %k:Minute:Second(?:\.Microseconds)?(?: ExYear)?
`_
```

```
Lines: 1 lines, 0 ignored, 1 matched, 0 missed
[processed in 0.01 sec]
```

**Listing 30.38** »fail2ban-regex«

In Listing 30.38 wird dem Programm `fail2ban-regex` als erster Parameter die Log-Zeile übergeben und als zweiter Parameter der reguläre Ausdruck. Anschließend verarbeitet das Programm die Daten und gibt Ihnen eine detaillierte Übersicht zurück.

Da nicht alle Log-Einträge auf allen Distributionen gleich sind, sollten Sie mit `fail2ban-regex` prüfen, ob die vordefinierten Ausdrücke auch zu Ihrem System passen.

## 30.8 OpenVPN

Ein VPN (*Virtual Private Network*) stellt eine sichere Verbindung über ein öffentliches Medium her. Trotz verschiedener Umsetzungen ist das Grundkonzept stets dasselbe: Um sicher Daten austauschen zu können, wird ein Tunnel erzeugt, in dem die Daten verschlüsselt werden, sodass ein Mitlesender nur den verschlüsselten Datentransfer sieht.

Jede VPN-Implementierung steht vor drei großen Problemen:

- ▶ **Autorisierung** – Wie wird die Kommunikation erlaubt?
- ▶ **Authentifizierung** – Wie weist sich ein Kommunikationspartner aus?
- ▶ **Integrität** – Sind die übertragenen Daten unversehrt?

Darüber hinaus gibt es zwei unterschiedliche Einsatzgebiete:

- ▶ **Roadwarrior**  
Außendienstmitarbeiter, die sich mit einer zentralen Firmeninfrastruktur verbinden, um ihre Daten abzugleichen oder Dienste zu nutzen
- ▶ **Site-to-Site**  
eine feste Verbindung zweier Standorte, die jeweils mit dem gegenüberliegenden Netzwerk kommunizieren sollen



Es gibt verschiedene Ansätze, um diese Probleme zu lösen. Aufgrund des Umfangs und der Komplexität der mathematischen Grundlagen verzichten wir an dieser Stelle auf eine ausführliche Erläuterung. Der Siegeszug von OpenVPN begann bereits beim Konzept. Als erste VPN-Implementierung setzt es auf eine Trennung der Netzwerkebene und der Verschlüsselung. Dieses Konzept vereinfacht die Umsetzung auf verschiedenen Betriebssystemen und



führt zugleich zu einer vereinfachten Konfiguration. Zusätzlich wurde OpenVPN mit *easy-rsa* eine Skriptsammlung an die Seite gestellt, die es auch dem ungeübten Nutzer erlaubt, sehr einfach eine Zertifikatsinfrastruktur zu betreiben (auch wenn diese mittlerweile als eigenständiges Paket gepflegt wird). All dies führte zur schnellen Akzeptanz und trug unweigerlich zur Beliebtheit von OpenVPN bei.

In diesem Abschnitt begleiten wir Sie von der Installation und der Einrichtung der Zertifikatsinfrastruktur über den Betrieb einer Site-to-site-Anbindung bis hin zur Einrichtung von Außendienstmitarbeitern. Damit Sie nicht den Überblick verlieren, haben wir an vielen Stellen Tabellen und Grafiken eingebunden, die Ihnen zum Beispiel die Verteilung der Zertifikate an die richtigen Standorte verdeutlichen.

### 30.8.1 Serverinstallation – OpenVPN, PKI und Co.

Die Installation kann aus den Paketquellen erfolgen. Installieren Sie die Pakete *openvpn* und *easy-rsa*. Bei CentOS müssen Sie die EPEL-Repositories hinzugefügt haben (`sudo dnf install epel-release`).

#### Betriebsarten

OpenVPN kann im *tun*- oder im *tap*-Modus verwendet werden. Diese beiden Betriebsarten unterscheiden sich eklatant voneinander:

- ▶ **tap (Bridging):** Es werden auch L2-Pakete durch den Tunnel geschleust. Dies ist vor allem dann notwendig, wenn auch andere Protokolle als TCP/IP durch den Tunnel verfügbar sein sollen (zum Beispiel IPX).
- ▶ **tun (Routing):** Pakete zwischen den beiden (oder mehreren) Endpunkten werden geroutet. Es werden keine L2-Pakete übertragen!

Selbstverständlich werden VPNs heutzutage nicht mehr über einen *PSK*<sup>8</sup> autorisiert, sondern über Zertifikate. Da die Erstellung und der Betrieb einer hierfür notwendigen Kopfstelle (PKI) äußerst komplex sind, wurde bei OpenVPN die Skriptsammlung *easy-rsa* hinzugefügt. Weitere Informationen zum Thema Zertifikate finden Sie in Kapitel 31, »Verschlüsselung und Zertifikate«.

Im Testumfeld (siehe Abbildung 30.2), das allen Konfigurationen zugrunde liegt, wird zur Einwahl das Transfernetz 10.2.1.0/24 verwendet. Der zentrale OpenVPN-Server erhält die erste IP-Adresse aus diesem Netz (.1). Der Zweigstelle Berlin wird die IP-Adresse 10.2.1.240 zugewiesen und der Zweigstelle Hamburg die IP-Adresse 10.2.1.250. Externe Mitarbeiter oder Fremdfirmen wählen sich im unteren Bereich des Class-C-Netzes ein. Dem Beispielclient »Max Mustermann« wird die IP-Adresse 10.2.1.3 zugewiesen.

<sup>8</sup> *Pre-sharded Key*, engl. für *vorher ausgetauschter Schlüssel*

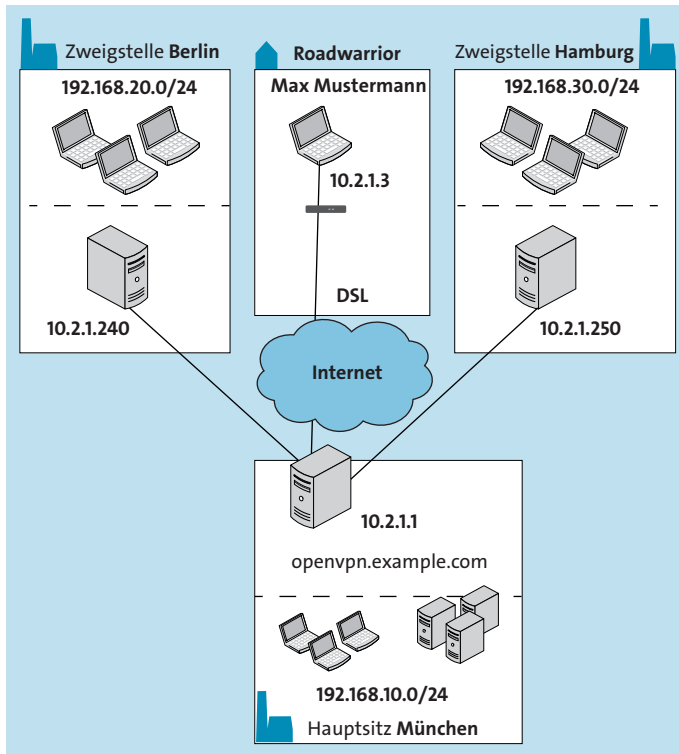


Abbildung 30.2 Netzwerkübersicht: OpenVPN-Testumfeld

Die aktuelle Version von OpenVPN 2.5 ist in allen gängigen Distributionen Bestandteil der Paketquellen. Während der Installation werden die folgenden Abhängigkeiten direkt mit installiert: *openssl* (Verschlüsselung) und *lzo* (Komprimierung). Skripte der Sammlung *easy-rsa* werden in den Distributionen unterschiedlich eingebunden und lassen sich daher leider sehr unterschiedlich bedienen, führen aber zum gleichen Ergebnis. Die Unterschiede stellen wir Ihnen nun genauer vor.

### Debian/Ubuntu: easy-rsa

Zunächst erstellen Sie ein eigenes Verzeichnis für die *Public Key Infrastructure* (kurz *PKI*). Führen Sie dafür den Befehl aus Listing 30.39 aus:

```
root@example:/etc/openvpn/# make-cadir ca
```

**Listing 30.39** Erstellen des PKI-Verzeichnisses unter »/etc/openvpn/ca«

Wechseln Sie anschließend in das neu erstellte Verzeichnis, und passen Sie die Datei *vars* an. Dort können Sie Standardwerte hinterlegen, die Ihnen bei der Erstellung der Zertifikate als Default angeboten werden. Passen Sie folgende Werte in der Datei an:

```
[...]
# Choose a size in bits for your keypairs. The recommended value is 2048. Using
# 2048-bit keys is considered more than sufficient for many years into the future.[...]
#set_var EASYRSA_KEY_SIZE      2048
[...]
# In how many days should the root CA key expire?
#set_var EASYRSA_CA_EXPIRE     3650
[...]

# In how many days should certificates expire?
#set_var EASYRSA_CERT_EXPIRE   825
[...]

# Organizational fields (used with 'org' mode and ignored in 'cn_only' mode.)
# These are the default values for fields which will be placed in the [...]
#set_var EASYRSA_REQ_COUNTRY   "US"
#set_var EASYRSA_REQ_PROVINCE  "California"
#set_var EASYRSA_REQ_CITY     "San Francisco"
#set_var EASYRSA_REQ_ORG      "Copyleft Certificate Co"
#set_var EASYRSA_REQ_EMAIL    "me@example.com"
#set_var EASYRSA_REQ_OU       "My Organizational Unit"
[...]
```

**Listing 30.40** Anpassung in »vars«

Der Parameter `EASYRSA_KEY_SIZE` gibt die Schlüssellänge der zu erstellenden Zertifikate an. Wie im Erläuterungstext bereits beschrieben wurde, verlangsamt ein Wert von 2048 zwar die TLS-Aushandlungszeit und die Generierung des DH-Schlüssels, der Wert sollte aber dennoch so »hoch« gesetzt werden, damit Ihre Infrastruktur auch über einen längeren Zeitraum wachsenden Anforderungen genügt.

Die Parameter `EASYRSA_CA_EXPIRE` und `EASYRSA_KEY_EXPIRE` geben die Gültigkeitsdauer der CA und der durch sie erstellten Zertifikate an (per Default zehn Jahre). In der freien Wirtschaft hat sich eine Gültigkeit von zehn bis zwölf Jahren für eine CA bewährt, für Clientzertifikate hingegen eine Gültigkeitsdauer von zwei bis vier Jahren. Passen Sie diese Werte Ihren Sicherheitsanforderungen an.

Falls Sie die Gültigkeitsdauer für Clientzertifikate auf nur ein Jahr verkürzen wollen, so denken Sie daran, dass Sie die Zertifikate dann auch jedes Jahr wechseln müssen!



Die im letzten Block aufgeführten Parameter geben die Daten Ihrer PKI an. Passen Sie diese entsprechend der nachstehenden Auflistung an:

- ▶ `EASYRSA_REQ_COUNTRY`  
Land als Kürzel (DE für Deutschland)

- ▶ EASYRSA\_REQ\_PROVINCE  
Region als Kürzel (in Deutschland die Bundesländer NRW, BY etc.)
- ▶ EASYRSA\_REQ\_CITY  
Ort, ausgeschrieben
- ▶ EASYRSA\_REQ\_ORG  
der Name des Unternehmens
- ▶ EASYRSA\_REQ\_EMAIL  
gültige E-Mail-Adresse, die bei Problemen oder Fragen erreicht werden kann

Nachdem Sie die Werte in der Datei `vars` angepasst haben, können Sie die PKI initialisieren. Dies erreichen Sie über die folgenden sprechenden Befehle:

```
root@example:/etc/openvpn/ca# ./easyrsa init-pki
```

```
init-pki complete; you may now create a CA or requests.
```

```
Your newly created PKI dir is: /etc/openvpn/ca/pki
```

#### **Listing 30.41** Erstellen der CA

Wenn Sie die Ausgabe aus Listing 30.41 erhalten, dann ist Ihre CA fast einsatzbereit. Nachdem dies erledigt ist, können Sie mit dem Erstellen der CA fortfahren:

```
root@example:/etc/openvpn/ca# ./easyrsa build-ca
```

```
Using SSL: openssl OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)
```

```
Enter New CA Key Passphrase: <PASSWORT>
```

```
Re-Enter New CA Key Passphrase: <PASSWORT>
```

```
You are about to be asked to enter information that will be incorporated  
into your certificate request.
```

```
What you are about to enter is what is called a Distinguished Name or a DN.
```

```
There are quite a few fields but you can leave some blank
```

```
For some fields there will be a default value,
```

```
If you enter '.', the field will be left blank.
```

```
-----
```

```
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:Example Ltd. CA
```

```
CA creation complete and you may now import and sign cert requests.
```

```
Your new CA certificate file for publishing is at:
```

```
/etc/openvpn/ca/pki/ca.crt
```

#### **Listing 30.42** Aufräumen und CA erstellen

Während der Erstellung werden Ihnen zwei Fragen gestellt: CA Key Passphrase und Common Name. Das bei der CA Key Passphrase vergebene Passwort müssen Sie bei jeder Interaktion

mit der CA eingeben, oder Sie rufen den Befehl mit dem Parameter `nopass` auf. Der Common Name bestimmt den Namen Ihrer CA.

An dieser Stelle wird Ihnen – falls Sie bereits mit älteren Versionen von EasyRSA gearbeitet haben – ein Unterschied aufgefallen sein: Es wurde *nur* der Common Name abgefragt! Was ist mit den übrigen Attributen? Standardmäßig läuft EasyRSA jetzt im Modus `cn_only`, was das Verhalten erklären sollte. An der Sicherheit ändert dies nichts. Falls Sie aber trotzdem lieber »vollständige« Zertifikate erstellen wollen, dann müssen Sie dafür in der `vars`-Datei lediglich den Parameter `EASYRSA_DN` von `cn_only` auf `dn` ändern. Erstellen Sie ein Serverzertifikat über die soeben eingerichtete CA. Hierfür bietet *easy-rsa* das Skript `build-server-full`, das Sie, gefolgt vom FQDN des Servers und dem Befehl `nopass` (ohne Passwort), aufrufen:

```
root@example:/etc/openvpn/pki# ./easyrsa build-server-full openvpn.example.com nopass
```

```
Note: using Easy-RSA configuration from: ./vars
```

```
Using SSL: openssl OpenSSL 1.1.1d 10 Sep 2019
Generating a RSA private key
.....+++++
writing new private key to '/etc/openvpn/ca/pki/easy-rsa-1263.xqpC50/tmp.R5ryla'
-----
Using configuration from /etc/openvpn/ca/pki/easy-rsa-1263.xqpC50/tmp.iF1p0b
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName          :ASN.1 12:'openvpn.example.com'
Certificate is to be certified until Apr 12 09:41:53 2025 GMT (825 days)

Write out database with 1 new entries
Data Base Updated
```

#### Listing 30.43 Generierung des Serverzertifikats – Default-Werte

Da es sich bei dem Zertifikat um ein Serverzertifikat handelt, wurde als CN der FQDN des Servers angegeben – passen Sie daher den Wert an Ihre Installation an. Anschließend finden Sie unterhalb des Verzeichnisses `pki` die Dateien für den Server:

- ▶ `./issued/openvpn.example.com.crt`  
das eigentliche Zertifikat
- ▶ `./reqs/openvpn.example.com.req`  
der für die Erstellung notwendige *signing request*
- ▶ `./private/openvpn.example.com.key`  
der private Schlüssel

### 30.8.2 CentOS/openSUSE Leap: easy-rsa

Bevor wir mit der Erstellung der PKI beginnen, müssen Sie noch ein paar kleine Vorbereitungen treffen. Führen Sie den nachstehenden Befehl als root-Benutzer auf dem jeweiligen System aus:

```
### CentOS
[root@centos ~]# cp -r /usr/share/easy-rsa/3/* /usr/local/sbin
[root@centos ~]# cp /usr/share/doc/easy-rsa/vars.example /usr/local/sbin/vars

### openSUSE Leap
leap:~ # cd /etc/easy-rsa
leap:/etc/easy-rsa # cp vars.example vars
```

#### Listing 30.44 Vorbereitungen

Standardmäßig läuft *easy-rsa* im Modus `cn_only`, daher werden nur Zertifikate mit dem Attribut *Common Name* erzeugt. An der Sicherheit ändert dies nichts. Falls Sie aber trotzdem lieber »vollständige« Zertifikate erstellen wollen, dann müssen Sie die Anpassungen aus Listing 30.45 umsetzen. Öffnen Sie dafür die Datei *vars* (unter CentOS `/usr/local/bin/vars` und unter openSUSE Leap `/etc/easy-rsa/vars`), und passen Sie diese Parameter an:

```
set_var EASYRSA_DN      "org"
[...]
set_var EASYRSA_REQ_COUNTRY    "DE"
set_var EASYRSA_REQ_PROVINCE  "NRW"
set_var EASYRSA_REQ_CITY      "Moers"
set_var EASYRSA_REQ_ORG       "Example Ltd."
set_var EASYRSA_REQ_EMAIL     "pki@example.com"
set_var EASYRSA_REQ_OU        "SysTec"
[...]
```

#### Listing 30.45 Standardwerte in »vars« festlegen

Damit geben Sie die Standardwerte bei der Zertifikatserstellung vor, sodass Sie nicht immer wieder die gleichen Daten eintippen müssen, und der Modus wird von `cn_only` auf `org` geändert. Zur Erstellung der PKI genügt das folgende Kommando:

```
root@example:~ # mkdir /etc/openvpn/ca && cd $_
root@example:/etc/openvpn/ca# easyrsa init-pki
```

Note: using Easy-RSA configuration from: `/etc/easy-rsa/vars`

```
init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /etc/openvpn/ca/pki
```

#### Listing 30.46 Fehlermeldung bei der Erstellung des ersten Server-Zertifikats

Nachdem die PKI erfolgreich erstellt wurde, müssen wir die CA erzeugen. Auch hierfür genügt wieder ein Kommando:

```
root@example:/etc/openvpn/ca# easyrsa build-ca nopass
[...]
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:Example Ltd. CA
```

CA creation complete and you may now import and sign cert requests.  
Your new CA certificate file for publishing is at:  
/etc/openvpn/ca/pki/ca.crt

**Listing 30.47** Erstellung der »CA«

Das Kommando `build-ca` erstellt die CA. Mit dem Parameter `nopass` weisen Sie *easy-rsa* an, einen privaten Schlüssel ohne Passwort zu erstellen. Anderenfalls müssten Sie bei jedem Zertifikat, das Sie mit dieser CA erstellen, immer das CA-Passwort eingeben. Falls der Sicherheitsgewinn Ihnen die zusätzliche Tipperei wert ist, lassen Sie einfach den Parameter `nopass` beim Aufruf weg.

**CentOS ignoriert nopass!**

Bei CentOS wird der Parameter bei der CA-Erstellung ignoriert, sodass Sie dennoch ein Passwort vergeben müssen!



Mit der frisch erstellten CA erzeugen wir nun das erste Server-Zertifikat mit dem Kommando `build-server-full`:

```
root@example:/etc/openvpn# easyrsa build-server-full openvpn.example.com nopass
[...]
writing new private key to '/etc/openvpn/ca/pki/easy-rsa-26112.ZRIjIz/tmp.lXjo6K'
-----
Using configuration from /etc/openvpn/ca/pki/easy-rsa-26112.ZRIjIz/tmp.0K7MaF
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName           :ASN.1 12:'openvpn.example.com'
Certificate is to be certified until Apr 12 12:16:18 2023 GMT (825 days)
```

Write out database with 1 new entries  
Data Base Updated

**Listing 30.48** Erstellung des ersten Server-Zertifikats

Nun finden Sie in den nachstehenden Unterverzeichnissen die soeben erzeugten Dateien für den Server:

- ▶ `/etc/openvpn/ca/pki/issued/openvpn.example.com.crt`  
das eigentliche Zertifikat
- ▶ `/etc/openvpn/ca/pki/reqs/openvpn.example.com.req`  
der für die Erstellung notwendige *signing request*
- ▶ `/etc/openvpn/pki/private/openvpn.example.com.key`  
der private Schlüssel

### 30.8.3 Gemeinsam weiter

Bleiben wir beim OpenVPN-Server. Erzeugen Sie für diesen einen DH-Schlüssel (Diffie-Hellman-Schlüssel), der für den gesicherten Schlüsselaustausch verwendet wird. Hierfür bietet *easy-rsa* das Skript *gen-dh* an:

```
root@example:/etc/openvpn/ca# ./easyrsa gen-dh      ### CentOS/openSUSE: ohne "./"
[...]
root@example:/etc/openvpn/ca# mv pki/dh.pem /etc/openvpn/dh2048.pem
```

**Listing 30.49** Diffie-Hellman-Schlüssel erstellen: »dh2048.pem«

Die Erstellung kann, je nach System, einige Zeit in Anspruch nehmen. Anschließend finden Sie im Verzeichnis `/etc/openvpn/ca/pki` die Datei *dh.pem*. Verschieben Sie diese (wie in Listing 30.49 dargestellt) nach `/etc/openvpn/dh2048.pem`. Erstellen Sie darüber hinaus einen *tls-auth key*. Hierfür verwenden wir direkt OpenVPN:

```
root@example:/etc/openvpn# openvpn --genkey secret tls-auth.key
```

**Listing 30.50** Generierung von »tls-auth.key«

Beim *tls-auth.key* handelt es sich um einen HMAC (*Keyed-Hash Message Authentication Code*) zur weiteren Absicherung der Kommunikation. Nun sind die Serverzertifikate und Schlüssel erstellt, und Sie können eine beliebige Anzahl von Clientzertifikaten erstellen. Hierfür bietet *easy-rsa* das Skript *build-client-full*, das Sie, gefolgt vom Zertifikatsnamen, aufrufen:

```
$ easyrsa build-client-full Max_Mustermann nopass
```

**Listing 30.51** Erstellung eines Client-Zertifikats für »Max Mustermann«

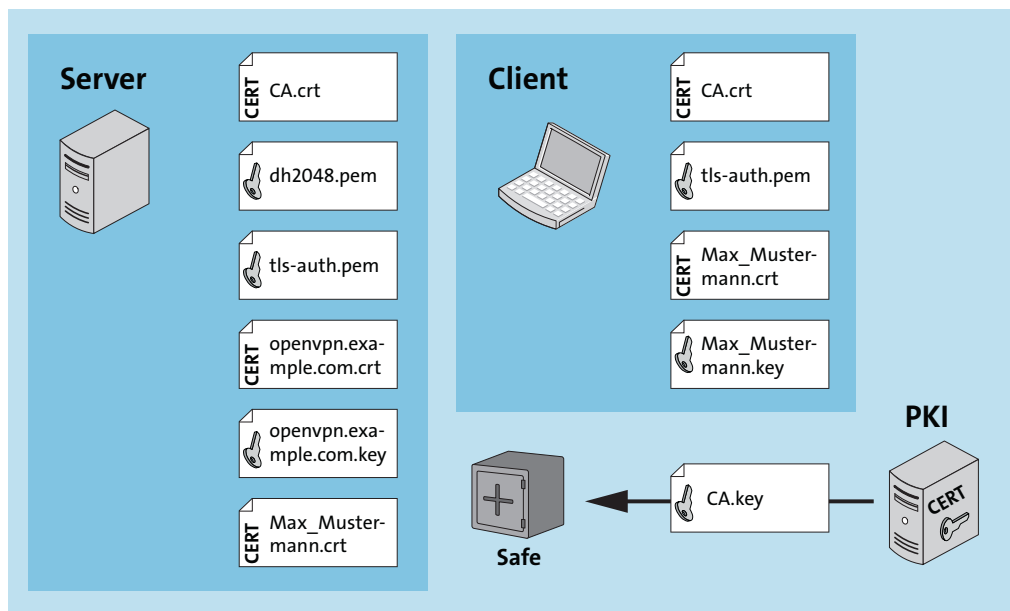
Der Zertifikatsname wird als *Common Name* verwendet. Prinzipiell können Sie einen beliebigen Namen für das Zertifikat vergeben. Aus Gründen der Übersichtlichkeit empfiehlt es sich aber, den Namen des Mitarbeiters oder des Systems zu verwenden, für das es erstellt wird. Wie Sie in Listing 30.51 sehen, kann auch dieses Zertifikat mit dem Parameter *nopass* ohne Passwort erzeugt werden.



| Dateiname               | Installationsort            | Zweck                    |
|-------------------------|-----------------------------|--------------------------|
| ca.crt                  | Server und alle Clients     | root-CA-Zertifikat       |
| ca.key                  | nur Server (besser im Safe) | root-CA-Schlüssel        |
| dh2048.pem              | nur Server                  | Diffie-Hellman-Schlüssel |
| tls-auth.pem            | Server und alle Clients     | HMAC                     |
| openvpn.example.com.crt | nur Server                  | Serverzertifikat         |
| openvpn.example.com.key | nur Server                  | Serverschlüssel          |
| Max_Mustermann.crt      | nur Client                  | Clientzertifikat         |
| Max_Mustermann.key      | nur Client                  | Clientschlüssel          |

**Tabelle 30.2** Übersicht über die Zertifikate

Die Verteilung der Zertifikate an die richtigen Orte stellt eine der größten Hürden bei der Implementierung von OpenVPN dar. Abbildung 30.3 und Tabelle 30.2 zeigen die richtige Verteilung der Zertifikate.



**Abbildung 30.3** Zertifikatsverteilung für einen Roadwarrior

### 30.8.4 Für den Roadwarrior

Nach der langwierigen Zertifikatserstellung geht es nun endlich an die eigentliche Konfiguration. Nach der Installation von OpenVPN auf dem Server finden Sie neben den *easy-rsa*-Skripten unter */etc/openvpn* nur noch die Verzeichnisse *client* und *server*, in denen die entsprechenden Konfigurationsdateien für die Verbindung als Client oder den Betrieb als Server abgelegt werden können. Bei openSUSE Leap ist dieses Verzeichnis hingegen leer – auf die notwendigen Anpassungen der Befehle weisen wir Sie entsprechend hin. Auf Debian-basierten Distributionen finden Sie in dem Verzeichnis ein Skript namens *update-resolver-conf*. Darauf werden wir später genauer eingehen.

#### Serverkonfiguration

Erstellen Sie zunächst das Verzeichnis */etc/openvpn/keys*. Dort platzieren Sie die notwendigen Zertifikate (entnehmen Sie diese bitte der Tabelle 30.2). Anschließend erstellen Sie eine Datei */etc/openvpn/server/server.conf* mit folgendem Inhalt:

```
port 443
dev tun
proto tcp
topology subnet
server 10.2.1.0 255.255.255.0
dh /etc/openvpn/keys/dh2048.pem
ca /etc/openvpn/keys/ca.crt
cert /etc/openvpn/keys/openvpn.example.com.crt
key /etc/openvpn/keys/openvpn.example.com.key
tls-auth /etc/openvpn/keys/tls-auth.key 0

keepalive 10 60
cipher AES-256-GCM
auth SHA256

persist-key
persist-tun
client-config-dir /etc/openvpn/ccd
ccd-exclusive

status /var/log/openvpn/status.log
log-append /var/log/openvpn/openvpn.log
verb 3
```

**Listing 30.52** »/etc/openvpn/server/server.conf«

Im ersten Block werden die Netzwerkparameter des OpenVPN-Servers gesetzt. Der Server wird auf dem TCP-Port 443 laufen und das *tun*-Verfahren anwenden. Durch die Angaben von

topology subnet und des nachstehenden Parameters server 10.2.1.0 255.255.255.0 wird den Tunneln ein privates Class-C-Netz zugewiesen. Der Server erhält in dieser Betriebsart immer die erste Adresse aus dem angegebenen Netzwerk – hier also die IP-Adresse 10.2.1.1.

Der zweite Block gibt den Speicherort der Zertifikate an. Die Angaben im dritten Block dienen der Benutzersteuerung. So gibt der Parameter `keepalive 10 60` an, dass alle zehn Sekunden ein Ping zur Gegenstelle abgesetzt wird. Antwortet die Gegenstelle 60 Sekunden lang nicht, wird der Tunnel zurückgesetzt, sodass eine Wiedereinwahl ermöglicht wird. Mit den Parametern `cipher` und `auth` wird der Verschlüsselungs- und die Signaturalgorithmus festgelegt.

Die Parameter `persist-key` und `persist-tunnel` weisen dem Benutzer immer den gleichen Tunnel zu. Das `ccd` (*client-configdir*) stellt eine zentrale Konfiguration dar, mit der eine 1:1-Zuordnung von Zertifikat zu IP-Adresse hergestellt werden kann. In großen Umgebungen wird ein OpenVPN-Server in der Regel in einer DMZ hinter zentralen Firewallsystemen betrieben. Da somit der gesamte Verkehr über die Firewalls geleitet wird, müssen explizite Regeln für den Teilnehmer erstellt werden. Der Parameter `client-config-dir` gibt das Verzeichnis der einzelnen Konfigurationen an. Mittels `ccd-exclusive` erlauben Sie nur solchen Benutzern (eigentlich Zertifikaten) den Zugriff, die auch über eine `ccd`-Konfiguration verfügen.

Im letzten Block werden die Pfade der Log-Dateien und deren Ausführlichkeit (*verbosity*) angegeben. Die serverseitige Konfiguration für den Benutzer *Max Mustermann* würde in der Datei `/etc/openvpn/ccd/Max_Mustermann` liegen und könnte wie folgt aussehen:

```
ifconfig-push 10.2.1.3 255.255.255.0
push "topology subnet"
push "redirect-gateway"
push "dhcp-option DNS 192.168.10.1"
push "dhcp-option WINS 192.168.10.1"
push "dhcp-option DOMAIN example.com"
```

**Listing 30.53** »/etc/openvpn/ccd/Max\_Musterman.conf«

### Zeitsteuerung mittels ccd

Wenn Sie die Direktive `ccd-exclusive` aktiviert haben, können Sie die Einwahl eines Nutzers einfach über ein `mv` der Konfigurationsdatei aus dem `ccd`-Verzeichnis heraus steuern. Über einen *cronjob* können Sie dies automatisiert abwickeln lassen. Und schon können externe Mitarbeiter oder Dienstleister sich nur noch in dem von Ihnen bestimmten Zeitfenster einwählen. Achtung: Bestehende Tunnel sind von dieser Einschränkung nicht betroffen!

OpenVPN bietet die sogenannte *Push*-Funktionalität. Damit können Sie Konfigurationen serverseitig festlegen. Diese Konfigurationen werden dann bei der Einwahl eines Clients direkt zugewiesen. Dies hat mehrere Vorteile: Zum einen werden fehlerhafte Konfigurationen durch den Anwender selbst unterbunden, und zum anderen kann ein Anwender durch das



Ändern seiner lokalen Konfiguration nicht an mehr Rechte gelangen – zum Beispiel durch das Ändern seiner IP-Adresse. Die zuvor beschriebenen Konfigurationen weisen dem Client eine feste IP-Adresse zu. Darüber hinaus wird der Client mittels *redirect-gateway* angewiesen, seine gesamte Kommunikation über den VPN-Tunnel zu routen.

Zusätzlich wird noch der DNS-, DHCP- und WINS-Server mitgeteilt. Diese Angaben verstehen leider nur Windows-Clients. Damit ein Linux-Client den internen DNS eingestellt bekommt, wird das bereits angesprochene Skript *update-resolv-conf* zur Verfügung gestellt. Auf die Besonderheiten werden wir in Abschnitt 30.8.5 im Unterabschnitt »Clientkonfiguration« näher eingehen. Aktivieren Sie den Kernelparameter *ip\_forward*, damit Ihr OpenVPN-Server Pakete weiterleiten kann, die nicht von ihm selbst stammen. Zur Laufzeit wird dies mit folgendem Befehl eingerichtet:

```
root@example:/# sysctl net.ipv4.ip_forward=1
```

**Listing 30.54** Aktivierung von »ip\_forward«

Dauerhaft ändern Sie diesen Kernelparameter in der Konfigurationsdatei */etc/sysctl.conf*. Dies ist wichtig, da bei der zuvor beschriebenen Variante die Einstellung bei jedem Neustart des Systems zurückgesetzt wird. Öffnen Sie daher die Datei mit root-Rechten, suchen Sie nach *net.ipv4.ip\_forward=1*, und entfernen Sie das Kommentarzeichen am Anfang der Zeile, oder ergänzen Sie die Zeile, falls diese nicht existiert. Ändern Sie unbedingt die Konfigurationsdatei ab, um den Kernelparameter dauerhaft zu ändern. Ansonsten laufen Sie Gefahr, nach einem Neustart, der vielleicht erst viele Monate später stattfindet, eine mühsame Fehlersuche durchführen zu müssen.

### 30.8.5 Start-Skript?

Falls Sie versuchen, Ihren soeben konfigurierten OpenVPN-Server wie üblich mit `systemctl start openvpn` zu starten, werden Sie eine Überraschung erleben. Unter Debian und Ubuntu erhalten Sie zwar die Anzeige, dass der Dienst läuft und alles in Ordnung ist, allerdings finden Sie keine Instanz. Unter CentOS und openSUSE Leap hingegen bekommen Sie die Fehlermeldung, dass es keinen Dienst `openvpn` gibt. Nun stellt sich die Frage: »Was soll das Ganze, und wie starte ich denn nun den Dienst?«

Dank `systemd` ist OpenVPN schlauer geworden. In älteren Versionen hat der Dienst einfach nach allen Dateien mit der Endung *conf* im Verzeichnis */etc/openvpn* Ausschau gehalten und versucht, diese zu starten. Somit konnte immer nur der gesamte OpenVPN-Prozess bedient werden und nicht einzelne Instanzen (mit eigenen Konfigurationen). Dies ist nun anders, da `systemd` standardmäßig eine Funktion zum Instanzieren mitliefert. Im Verzeichnis */lib/systemd/system/* (bei CentOS und openSUSE Leap mit vorangestelltem */usr*) finden Sie das Instanz-Template *openvpn-server@.service* (bei openSUSE Leap nur ein Template für alle *openvpn@.service* – passen Sie daher die nachstehenden Befehle entsprechend an). Um

nun die erstellte Konfiguration (*server.conf*) starten zu können, müssen Sie einfach einen passenden symbolischen Link anlegen:

```
root@example:/# systemctl -f enable openvpn-server@server.service
```

**Listing 30.55** OpenVPN-systemd-Start-Skript erzeugen

Mit dem Kommando aus Listing 30.55 wird die Konfigurationsdatei *server.conf* zum Starten vorgesehen – falls Sie einen anderen Namen verwendet haben, zum Beispiel *Meine-OpenVPN-Serverkonfiguration.conf*, dann müsste der Befehl entsprechend angepasst werden: `systemctl -f enable openvpn-server@MeineOpenVPN-Serverkonfiguration.service`.

Nun ist die Serverkonfiguration abgeschlossen. Wenn Sie OpenVPN mit dem soeben erstellten systemd-Start-Skript starten, wird der Dienst im Hintergrund ausgeführt. Gerade bei Fehlern oder nicht erklärbarem Verhalten empfiehlt es sich aber, OpenVPN in einer Konsole zu starten, sodass Sie (bei entsprechend hoch gesetzter *verbosity*) direkt eine umfangreiche Ausgabe erhalten. Setzen Sie hierfür den folgenden Befehl ab:

```
root@example:/# openvpn --config /etc/openvpn/server/server.conf
```

**Listing 30.56** OpenVPN in einer Konsole

#### Im Vordergrund nur ohne konfigurierte Log-Dateien

Beachten Sie, dass die Ausgabe auf der Konsole nur erfolgt, wenn Sie nicht die Direktive `log-append` in der Konfigurationsdatei gesetzt haben. Anderenfalls startet OpenVPN auch auf der Konsole im Hintergrund.



### Clientkonfiguration

Die Konfiguration von OpenVPN ist auf Windows- und Linux-Clients fast identisch. Lediglich einige kleine Punkte wie die Pfadangaben oder die bereits beschriebene *Push*-Direktive verhalten sich unterschiedlich. Laden Sie den OpenVPN-Windows-Client unter <http://openvpn.net> herunter. Dort werden Ihnen zwei Versionen angeboten: zum einen der *Community-Client*, der als Open Source angeboten wird, und zum anderen der *openvpn.net*-Client, für den Sie auch Support erwerben können. Nach der Installation legen Sie unter `C:\Programme\OpenVPN` den Ordner *keys* an. Kopieren Sie die in Tabelle 30.2 beschriebenen Zertifikate in den neu angelegten Ordner. Die Konfiguration unserer Beispielumgebung könnte auf einem Windows-Client wie folgt aussehen:

```
remote openvpn.example.com 443
dev tun
proto tcp
client
remote-cert-tls server
```

```

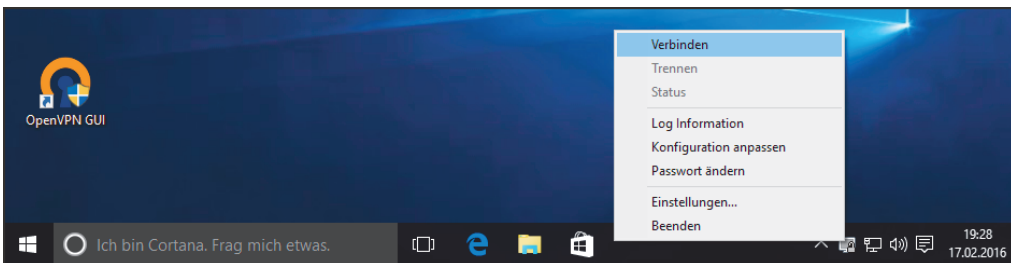
tun-mtu 1500
ca "C:\\Programme\\OpenVPN\\keys\\ca.crt"
cert "C:\\Programme\\OpenVPN\\keys\\Max_Mustermann.crt"
key "C:\\Programme\\OpenVPN\\keys\\Max_Mustermann.key"
tls-auth "C:\\Programme\\OpenVPN\\keys\\tls-auth.key" 1
key-direction 1

cipher AES-256-GCM
auth SHA256
verb 3

```

**Listing 30.57** Windows: »client.ovpn«

Wie bereits bei der Serverkonfiguration ist die Datei *client.ovpn* in drei Blöcke unterteilt. Im ersten Block wird wieder die Netzwerkkonfiguration vorgenommen. Die Direktive *client* gibt hierbei an, dass es sich um einen Client handelt. Der zweite Block gibt die Speicherorte der Zertifikate an. Bei den doppelten zurückgelehnten Schrägstrichen (*Backslashes*) handelt es sich nicht um einen Druckfehler: Die OpenVPN-Syntax erwartet die Pfadangaben in diesem Format. Im letzten Block wird die Verschlüsselung angegeben und das Log-Level auf die dritte Stufe gestellt. Nun können Sie die *OpenVPN GUI* starten und über einen Rechtsklick auf das OpenVPN-Symbol im Systemtray so, wie in Abbildung 30.4 gezeigt, den Tunnel aufbauen.



**Abbildung 30.4** OpenVPN-Windows-Client

Auf Linux-Clients installieren Sie ebenfalls das Paket *openvpn* wie auf dem Server. Legen Sie das Verzeichnis */etc/openvpn/keys* an, und kopieren Sie die benötigten Zertifikate (siehe Tabelle 30.2) hinein. Anschließend legen Sie die Datei */etc/openvpn/client/client.conf* mit folgendem Inhalt an:

```

remote openvpn.example.com 443
dev tun
proto tcp
client
remote-cert-tls server
tun-mtu 1500

```

```
ca /etc/openvpn/keys/ca.crt
cert /etc/openvpn/keys/Max_Mustermann.crt
key /etc/openvpn/keys/Max_Mustermann.key
tls-auth /etc/openvpn/keys/tls-auth.key
```

```
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf
```

```
comp-lzo
verb 3
```

**Listing 30.58** Linux (Debian): »client.conf«

Wie bereits erläutert wurde, verstehen Linux-Clients die über *ccd* mitgeteilten DHCP-Optionen nicht. Je nach Distribution müssen Sie nun unterschiedliche Methoden verwenden, um ans Ziel zu gelangen. Unter Debian können Sie das mitgelieferte *update-resolv-conf*-Skript verwenden. Dieses wird über die Direktiven *up* und *down* eingebunden (siehe Listing 30.58).

Dabei handelt es sich um sogenannte *scripting hooks* (engl. für *Skript-Haken*), die Punkte darstellen, an denen OpenVPN es Ihnen ermöglicht, eigene Skripte zu hinterlegen und an bestimmten Stellen der Abarbeitung auszuführen. Dieses Skript wertet den *ccd*-Parameter *dhcp-option* aus und setzt die Datei */etc/resolv.conf* beim Tunnelauf- und -abbau entsprechend um. Dafür muss das Paket *resolvconf* installiert sein.

Selbstverständlich können Sie hier auch eigene Skripte hinterlegen, die für Sie zum Beispiel Netzwerkfreigaben einbinden oder andere Systemparameter für den Tunnel anpassen. Unter Ubuntu, bei dem die DNS-Konfiguration von *systemd-resolv* übernommen wird, müssen Sie die Zeilen wie folgt anpassen und das Paket *openvpn-systemd-resolved* installieren:

```
script-security 2
up /etc/openvpn/update-systemd-resolved
down /etc/openvpn/update-systemd-resolved
down-pre
```

**Listing 30.59** Anpassungen der »client.conf« unter Ubuntu

Auf OpenSUSE-Leap-Systemen müssen Sie hingegen Hand anlegen. Installieren Sie zunächst das Paket *openresolv*, und laden Sie anschließend das Skript *update-resolv-conf* aus dem GitHub-Repository <https://github.com/masterkorp/openvpn-update-resolv-conf/> herunter. Nachdem Sie der heruntergeladenen Datei noch Ausführungsrechte gegeben haben, genügt die Konfiguration aus Listing 30.58. Unter CentOS müssen Sie noch einen Schritt weiter gehen. Dort wird das Paket *openresolv* nämlich leider nicht in den Repositories angeboten. Laden Sie daher das Programm (das im Wesentlichen eine Skriptsammlung darstellt) von der Projektseite herunter: <https://roy.marples.name/downloads/openresolv/>

Nach dem Entpacken genügt zur Installation der übliche Dreisprung mit den Befehlen `./configure && make && make install`. Nun müssen Sie sowohl auf openSUSE Leap- als auch auf CentOS-Systemen noch das Skript `update-resolv-conf` aus dem GitHub-Repository <https://github.com/masterkorp/openvpn-update-resolv-conf/> herunterladen und die Shellskripte unter `/etc/openvpn/` ablegen. Anschließend genügt auch hier die Konfiguration aus Listing 30.58. Auf Linux-Clients können Sie den Tunnel über die mitgelieferten `systemd`-Skripte starten (vorher anlegen mit: `systemctl -f enable openvpn-client@client.service`) oder aus der Konsole mit dem folgenden Befehl:

```
root@example:/# openvpn --config /etc/openvpn/client/client.conf
```

**Listing 30.60** OpenVPN in einer Konsole

Bei openSUSE Leap müssen Sie den Befehl `systemctl` erneut ohne das Suffix `-client` absetzen.

### 30.8.6 Site-to-site

Die Anbindung einer oder mehrerer Außenstellen ist ebenfalls ein Schwerpunkt bei VPNs. Dies können Sie mit OpenVPN selbstverständlich auch konfigurieren. Erstellen Sie zunächst für die Zweigstellen Berlin und Hamburg ein Clientzertifikat über die PKI des Hauptsitzes – mit der wir bisher auch gearbeitet haben. Dies können Sie über das *easy-rsa*-Skript `build-key` erreichen:

```
root@example:/etc/openvpn/ca# ./easyrsa build-client-full berlin nopass
[...]
root@example:/etc/openvpn/ca# ./easyrsa build-client-full hamburg nopass
[...]
```

**Listing 30.61** Clientzertifikate der Zweigstellen (unter CentOS/openSUSE Leap ohne »./«)



#### Clientzertifikate

Auch wenn es sich bei den OpenVPN-Servern der Zweigstellen um Server handelt, werden diese als »Clients« behandelt. Erstellen Sie daher unbedingt Clientzertifikate über das *easy-rsa*-Skript `build-client-full`.

Da die Serveranbindung der Zweigstellen in der Zentrale von OpenVPN als Clientanbindung gehandhabt wird, müssen auch die entsprechenden Konfigurationen hinterlegt werden. Erstellen Sie daher für die Zweigstellen jeweils eine Konfigurationsdatei im `ccd`:

```
ifconfig-push 10.2.1.240 255.255.255.0
push "topology subnet"
push "redirect-gateway"
```



```
# Angabe des hinter dem Client befindlichen Netzes:
iroute 192.168.20.0 255.255.255.0
```

**Listing 30.62** »/etc/openvpn/ccd/berlin«

Die Konfiguration entspricht der des bereits eingerichteten Roadwarriors *Max Mustermann*. Lediglich die Erweiterung um die `iroute`-Direktive wurde hinzugefügt. Diese Direktive weist OpenVPN an, das angegebene Netz als »hinter diesem Client befindlich« zu betrachten. Genauso erstellen wir die `ccd`-Konfiguration für die Zweigstelle Hamburg:

```
ifconfig-push 10.2.1.250 255.255.255.0
push "topology subnet"
push "redirect-gateway"
```

```
# Angabe des hinter dem Client befindlichen Netzes:
iroute 10.254.9.0 255.255.255.0
```

**Listing 30.63** »/etc/openvpn/ccd/hamburg«

Zusätzlich muss die zentrale OpenVPN-Konfiguration angepasst werden. Ergänzen Sie folgende Zeilen, um die Konfiguration des zentralen OpenVPN-Servers abzuschließen:

```
# Route auf dem openVPN-Server selbst:
route 192.168.20.0 255.255.255.0 10.2.1.240 0
route 192.168.30.0 255.255.255.0 10.2.1.250 0
client-to-client
```

```
# Routen an die Clients weitergeben:
push "route 192.168.20.0 255.255.255.0"
push "route 192.168.30.0 255.255.255.0"
```

**Listing 30.64** Zentrale: »/etc/openvpn/server/server.conf«

Durch die Direktive `route` wird dem OpenVPN-Server selbst mitgeteilt, dass nach der Einwahl der jeweiligen Clients die angegebenen Routen (lokal) hinzugefügt werden müssen. Der syntaktische Aufbau der Direktive ist dabei `route <Netzwerk> <Netzmaske> <Metrik>`. Die Direktive `client-to-client` lässt die Kommunikation zwischen den Zweigstellen und dem Roadwarrior zu. Ohne diese könnten die Einwählenden sich untereinander nicht erreichen. Die letzten beiden Konfigurationszeilen verteilen die Routing-Informationen an die Clients selbst. Dabei kommt der eigentliche Nutzen der in den `ccd`-Konfigurationen hinterlegten `iroute`-Direktive zum Tragen. Diese ist nämlich dafür verantwortlich, dass zum Beispiel dem Client der Zweigstelle Berlin nur die Route »192.168.30.0 via 10.2.1.1« nach der Einwahl zugewiesen wird. Anderenfalls würde dem OpenVPN-Server in der Zweigstelle nämlich auch die Route seines eigenen Netzes mitgeteilt, und dieser würde die Pakete in Richtung Zentrale routen, was eine Kommunikation unmöglich machen würde.



Abschließend fehlt noch eine der wichtigsten und leider auch am häufigsten vergessenen Konfigurationen: das Routing auch zu aktivieren!

Wie bereits beim zentralen OpenVPN-Server müssen die Server der Zweigstellen in die Lage versetzt werden, Pakete zu routen, die nicht vom Server selbst stammen. Aktivieren Sie daher, wie in Listing 30.54 beschrieben, das `ip_forward`.

### 30.8.7 Simple-HA

Ausfallsicherheit gewinnt in der heutigen Zeit immer mehr an Bedeutung. Daher beherrschen viele Dienste von Haus aus Mechanismen, um für HA<sup>9</sup> zu sorgen. So auch Open-VPN, zumindest für reine Clientumgebungen, hinter denen sich nicht weitere Netze verbergen (zum Beispiel Roadwarriors). Neben der reinen Konfiguration von VPN-HA muss hierbei aber stets die Netzwerkkonfiguration angepasst werden. Planen Sie dies im Vorfeld mit ein.

Die Netzwerkkomponenten, die sich hinter Ihren VPN-Servern befinden, müssen wissen, zu welchem Endgerät die Netze geroutet werden müssen. Betreiben Sie nun zwei VPN-Server, da Sie ja Hochverfügbarkeit erreichen wollen, müssen Sie den Systemen unterschiedliche Netze zuweisen – und dementsprechend auch den Clients dahinter, damit die Netzwerkrouuten konsistent bleiben können.

#### DNS-basierte HA

Falls Sie bereits eine funktionierende OpenVPN-Infrastruktur in Betrieb haben, diese aber um HA erweitern wollen, ohne Änderungen an den Clients vorzunehmen, bietet sich die DNS-basierte HA-Lösung an. Hierbei müssen Sie in Ihrem DNS lediglich den A-Record Ihres OpenVPN-Servers doppelt anlegen und in die jeweiligen IP-Adressen auflösen lassen. Das DNS löst dann im Round-Robin-Verfahren entweder die eine oder die andere IP-Adresse auf.

Dies geschieht für die Clients transparent, sodass Sie »nur« die Serverinfrastruktur anpassen müssen. Kopieren Sie dafür einfach den gesamten Verzeichnisinhalt von `/etc/openvpn` des ursprünglichen OpenVPN-Servers auf den neuen, und passen Sie die Netzwerkangaben an. Falls Sie `ccd` einsetzen, müssen Sie selbstverständlich in jeder Clientkonfiguration auch die neuen IP-Adressangaben ändern.

#### Konfigurierte HA

Falls Sie mehr Kontrolle darüber benötigen, auf welchen Servern sich die Clients verbinden, können Sie diese mit ein paar einfachen Mitteln direkt im OpenVPN konfigurieren. Klonen Sie zunächst, wie bei DNS-basierter HA, die Serverkonfiguration. Geben Sie den Servern aber eigenständige Namen im DNS.

---

<sup>9</sup> *High Availability*, engl. für *Hochverfügbarkeit*

Anschließend können Sie auf der Clientseite mehrere *remote*-Zeilen hinterlegen. Diese werden von OpenVPN in ihrer Reihenfolge abgearbeitet. Falls ein Server nicht erreicht werden kann, wird der nächste verwendet. Je nach Sortierung der Zeilen haben Sie hier die erste Möglichkeit, eine Verteilung vorzunehmen. Wenn Sie zum Beispiel zwei OpenVPN-Server (*openvpn1.example.com* und *openvpn2.example.com*) betreiben, die jeweils vorrangig für interne und externe Benutzer verwendet werden sollen, sich aber im Fehlerfall vertreten sollen, dann können Sie bei der Clientkonfiguration der internen Benutzer den ersten Server an erster Stelle setzen und bei den externen Benutzern an zweiter.

Falls Sie eher eine Lastverteilung mit integrieren möchten, können Sie clientseitig auch den Parameter *remote-random* setzen. Dieser weist den Client an, einen beliebigen Server aus den zur Verfügung stehenden *remote*-Zeilen zu verwenden.

### Hartnäckigkeit

OpenVPN ist von sich aus sehr hartnäckig. Das heißt, dass erst dann der nächste Server (*remote*-Zeile) verwendet wird, wenn der erste Server sehr oft nicht geantwortet/reagiert hat. Um einen Serverwechsel zu beschleunigen, können Sie auf der Clientseite den zusätzlichen Parameter *server-poll-timeout N* setzen. Dabei gibt *N* die Anzahl der Versuche an, bis der nächste Server verwendet wird.



### 30.8.8 Tipps und Tricks

Beim Betrieb einer OpenVPN-Umgebung werden Ihnen an der einen oder anderen Stelle Merkwürdigkeiten auffallen. Die Logs von OpenVPN sind dankenswerterweise sehr ausführlich, an der einen oder anderen Stelle aber leider auch irreführend. In diesem Abschnitt wollen wir Ihnen einige unserer Erfahrungen mit auf den Weg geben, damit Sie die eine oder andere Stolperfalle gekonnt umgehen können.

#### register-dns

Einige Windows-Clients setzen die von OpenVPN übergebenen DNS/DHCP-Optionen nicht korrekt um oder aktivieren diese nur zum Teil. Um dies zu umgehen, wurde der Parameter *register-dns* hinzugefügt. Setzen Sie diesen auf Windows-Clients, wird nach dem Tunnelaufbau der DNS-Clientdienst des Systems nochmals angewiesen, auch wirklich die neuen DNS/DHCP-Optionen zu aktivieren.

### Standardinstallationspfad

Leider bleibt auch OpenVPN von Bugs nicht verschont. Falls Ihre Windows-Installation nicht im Standardpfad *C:\Windows* vorgenommen wurde, greift der Parameter *register-dns* leider zurzeit ins Leere. Der Pfad wurde nämlich hart einkodiert.



## DNS-Leak

Ab Windows 10 hat Microsoft eine Neuerung eingeführt: Es verwendet jeden erreichbaren DNS-Server – egal, ob Sie einen VPN-Tunnel aufgebaut haben oder nicht. Dies führt zu dem hitzig diskutierten *DNS Leak*, da durch diese Neuerung das Unterwandern eines Systems vereinfacht wird. Praktischerweise wurde der Workaround *openvpn-fix-dns-leak-plugin* des Entwicklers *ValdikSS* in OpenVPN mit aufgenommen. Dieses Plug-in sorgt dafür, dass nach dem Tunnelaufbau mittels der Windows-Firewall der Zugriff auf DNS-Server außerhalb des Tunnels unterbunden wird. Dafür muss in der Client-Konfiguration lediglich die Direktive `block-outside-dns` gesetzt sein. Für Windows 7 bis 8.1 müssen Sie diese Direktive auskommentieren.

Und da Linux Windows in nichts nachsteht, wurde der DNS-Leak in alle Systeme mit übernommen, die mit *systemd-resolv* arbeiten (z. B. Ubuntu). Dieses kontroverse Thema wird sehr leidenschaftlich diskutiert. Fakt ist: In der Regel möchte man als VPN-Administrator sicherstellen, dass verbundene Systeme sämtliche Kommunikation über den Tunnel leiten. Um dies auch bei Linux-Clients mit *systemd-resolv* zu erreichen, müssen Sie in der *ccd* des jeweiligen Clients nur die Zeile `dhcp-option DOMAIN-ROUTE .` hinzufügen. Diese sorgt dafür, dass der von OpenVPN mitgeteilte DNS stets die höchste Priorität hat und somit einzig und allein verwendet wird.

## explicit-exit-notify

In OpenVPN-Umgebungen, die auf UDP laufen (Serverparameter `proto udp`), wird das Abmelden eines VPN-Tunnels vom Server nicht sofort festgestellt. Erst nach einer gewissen Zeit wird der belegte Slot wieder freigeräumt – dabei kann es unter Umständen zu Problemen bei der Wiedereinwahl kommen. Um dies zu unterbinden, wurde der Parameter `explicit-exit-notify` hinzugefügt. Dieser veranlasst den Client, beim Herunterfahren des Tunnels eine Meldung an den Server zu senden. Dies ist vor allem dann wichtig, wenn Sie mittels Skript-Hooks eingebundene Auswertungsskripte betreiben möchten.

## Windows-Installationspfad

Falls Sie nicht wie gewöhnlich Ihre Windows-Installation unter `C:\Windows` abgelegt haben, müssen Sie dies OpenVPN mitteilen. Anderenfalls kann OpenVPN die benötigten Programme nicht finden, was im Log entsprechend quittiert wird. Hierfür nutzen Sie einfach den Parameter `win-sys N`. Dabei spezifiziert `N` den Pfad zur Windows-Installation.



Beachten Sie, dass die Pfadangabe wieder mit doppelten zurückgelehnten Schrägstrichen (*Backslashes*) vorgenommen wird.

## Windows-Routing/Netzwerk

Windows bietet prinzipiell zwei Möglichkeiten an, wie die Netzwerksystemeinstellungen verändert werden können: zum einen über eine spezielle API und zum anderen über die Pro-

gramme, die Sie auch in einer *cmd*-Shell verwenden können. Bei Problemen mit dem Setzen von IP-Adressen oder Netzwerkrouuten hilft es, OpenVPN anzuweisen, nicht die standardmäßig gesetzte Methode *adaptive* zu verwenden, sondern das Pendant über die Windows-Programme. Setzen Sie dafür auf der Clientseite die Parameter *ip-method netsh* und *route-method exe*. Durch diese Methoden erhalten Sie unter Umständen eine genauere Vorstellung von dem Problem aufgrund der erweiterten Fehlerausgabe.

### Modemverbindungen (DSL-Modem/UMTS)

Bei Modemverbindungen werden die Routing-Informationen nicht auf ein Gateway, sondern auf eine Verbindung gesetzt. Dies ist zum Beispiel bei DSL-Modemverbindungen, die Sie mittels DFÜ-Einwahlverbindung eingerichtet haben, oder bei einigen UMTS-Verbindungen der Fall. In solch einer Umgebung kann die Default-Route durch OpenVPN nicht korrekt gesetzt werden, wenn Sie serverseitig den Parameter *redirect-gateway* gesetzt haben.

Dafür bietet Ihnen OpenVPN aber einen kleinen Trick an: Die Default-Route wird mithilfe der Parametererweiterung *redirect-gateway def1* halbiert. Sie finden auf solchen Systemen dann sowohl eine vollständige Default-Route, die auf die Verbindung zeigt, als auch zwei halbe Routen für *0.0.0.0/1* und *128.0.0.0/1*.

Aufgrund des Routing-Konzepts, das immer enger gefasste Routen bevorzugt, wird der gesamte Verkehr über den Tunnel geleitet – bis auf die jeweiligen Netzwerk- und Broadcast-Routen, die an diesen Stellen aber keine Hürde darstellen.

### Debug

Um Fehlern schnell auf die Schliche zu kommen, hilft es, die Verbosity schrittweise zu erhöhen. Dies sollten Sie stets sowohl auf dem Server als auch auf dem Client vornehmen, da Sie nur so die Gesamtheit überblicken können. Übertreiben Sie es allerdings nicht mit der Verbosity, da das Log ansonsten schnell sehr unübersichtlich wird. Auf Windows-Clients sollten Sie auch in Betracht ziehen, die Methoden zur Einstellung von IP-Adressen und Netzwerkrouuten umzustellen. Die Windows-eigenen Programme haben zum Teil eine andere oder eine ausführlichere Ausgabe, wodurch Sie gegebenenfalls schneller auf den Fehler stoßen.

Es gibt aber (leider) auch Fehler, die sich nicht direkt vom Log ableiten lassen. Damit Ihnen folgende Phänomene keine Kopfschmerzen bereiten, haben wir hier einige Lösungsansätze für Sie aufgelistet:

#### ► Falsch verbunden – Verbindung mit dem falschen Port

Wenn Sie auf einem Server mehrere OpenVPN-Instanzen betreiben, laufen diese auf unterschiedlichen Ports. Wenn sich nun ein Client der ersten Instanz mit dem Port der zweiten verbindet, kommt zwar eine Verbindung zustande, Sie können das System allerdings nicht erreichen. Das ist auf die falschen Netzwerkrouuten zurückzuführen. Achten Sie also penibel darauf, welcher Client welche Konfigurationsdatei bekommt!

- ▶ **Einige Anwendungen laufen im Tunnel, andere nicht – Tunnelabbrüche**  
Falls Ihnen dieses Phänomen begegnet, verkleinern Sie zunächst die MTU-Größe mithilfe des Clientparameters `tun-mtu`. Nicht alle Programme füllen die Netzwerkpakete zur Gänze, daher kann es vorkommen, dass ein Programm läuft, ein anderes aber nicht.
- ▶ **Nach einem Tunnelabbruch kann keine erneute Verbindung aufgebaut werden**  
Einige (DSL-)Router halten Verbindungen offen, obwohl diese beendet wurden. Lassen Sie daher den lokalen Router durchstarten. Falls das Phänomen öfter auftritt, sollten Sie ein Firmware-Upgrade durchführen lassen.
- ▶ **Der Tunnel funktioniert nur für ein paar Minuten**  
Gerade auf Windows-Systemen mit installiertem Netzwerkmanager kann es vorkommen, dass die Routing-Informationen nach einer gewissen Zeit durch diese Programme geändert werden. Deinstallieren Sie nach Möglichkeit den Netzwerkmanager.
- ▶ **OpenVPN meldet: »tun-/tap-Device nicht gefunden«**  
Obwohl Sie in der Netzwerkübersicht von Windows den TUN-TAP-Adapter sehen können, wird bei einem Tunnelaufbau dieser Fehler angezeigt. Dies kann darauf zurückzuführen sein, dass die interne Zuordnung von Windows nicht mehr korrekt ist. Das kommt häufig vor, wenn auf dem Client zusätzliche Virtualisierungssoftware installiert wurde. Um dies zu umgehen, können Sie den Adapter neu installieren, ohne die ganze Installation erneut durchführen zu müssen. Im Installationsverzeichnis von OpenVPN (Standard: `C:\Programme\OpenVPN`) finden Sie im Ordner *bin* die Batchdateien *deltapall* und *addtap*, die dies für Sie übernehmen können.
- ▶ **Der Tunnel steht, es werden aber keine Daten übertragen**  
Prüfen Sie die lokalen Routen: unter Umständen wurde auf dem Client eine statische Route angelegt, die sich mit dem von Ihnen verwendeten Netzwerkbereich überschneidet. Dies kann durch die Installation einer Virtualisierungssoftware erfolgt sein – ändern Sie gegebenenfalls den entsprechenden Bereich in der Virtualisierungssoftware.

## 30.9 Schnell, Modern, Sicher: WireGuard

Seit einiger Zeit gibt es einen neuen Herausforderer in der Arena – die Rede ist natürlich von *WireGuard*. Das von *Jason A. Donenfeld* entwickelte Programm (und Protokoll) stellt eine der wenigen »echten« Neuerungen im VPN-Bereich der letzten Jahre dar. Im Gegensatz zu anderen VPN-Implementierungen (z. B. dem bereits vorgestellten *OpenVPN* oder *IPsec*) ist diese VPN-Technik direkt im Kernel integriert, was WireGuard einen enormen Geschwindigkeitsvorteil bringt. Zusätzlich ist es äußerst schlank implementiert.

In diesem Abschnitt zeigen wir Ihnen, wie schnell Sie mit WireGuard einen VPN-Tunnel aufsetzen können. Wir erörtern allerdings auch die Nachteile, die es mit sich bringt.

### 30.9.1 Schnell einen Tunnel einrichten

Mal schnell einen Tunnel mit WireGuard aufbauen: kein Problem! Sie benötigen weniger als zehn Zeilen in Ihrer Kommandozeile pro System, und Ihr Tunnel steht. Das glauben Sie nicht? Sehen Sie selbst! Wir gehen von folgendem Testszenario aus:

► **1. Teilnehmer:**

- FQDN: tom.example.com
- Externe-IP: 198.51.100.1
- VPN-IP: 10.0.0.1

► **2. Teilnehmer:**

- FQDN: jerry.example.com
- Externe-IP: 203.0.113.2
- VPN-IP: 10.0.0.2

30

Zunächst installieren Sie WireGuard auf Ihrem System so wie in Listing 30.65:

```
# Debian/Ubuntu
apt install wireguard

# openSUSE
$ sudo zypper addrepo https://download.opensuse.org/repositories/home:nandcd/\
openSUSE_Leap_15.0/home:nandcd.repo
$ sudo zypper install wireguard wireguard-tools

# CentOS
dnf install -y epel-release
dnf copr enable jdoss/wireguard
dnf install wireguard-tools
```

**Listing 30.65** Installation von WireGuard

Auf beiden Systemen müssen Sie nun die Befehle aus Listing 30.66 absetzen. Selbstverständlich müssen Sie die jeweiligen Schlüssel auf dem korrespondierenden System hinterlegen – beachten Sie daher bitte die jeweiligen Überschriften.

```
root@tom:~# umask 077
root@tom:~# wg genkey > private
root@tom:~# wg pubkey < private
F+Nk[...]imw=
root@tom:~# ip link add wg0 type wireguard
root@tom:~# ip addr add 10.0.0.2/24 dev wg0
root@tom:~# wg set wg0 private-key ./private
root@tom:~# ip link set wg0 up
```

```
### Ausgabe auf tom.example.com          ### Ausgabe auf jerry.example.com
root@tom:~# wg                          # root@jerry:~# wg
interface: wg0                          # root@jerry:~# interface: wg0
  public key: F+Nk[...]imw=            # root@jerry:~#   public key: F+Nk[...]imw=
  private key: (hidden)                # root@jerry:~#   private key: (hidden)
  listening port: 39478                # root@jerry:~#   listening port: 34743

### tom.example.com
root@tom:~# wg set wg0 peer qMka[...]3R0= allowed-ips 10.0.0.2/32 \
  endpoint 203.0.113.2:34743

### jerry.example.com
root@jerry:~# wg set wg0 peer F+Nk[...]imw= allowed-ips 10.0.0.1/32 \
  endpoint 198.51.100.1:39478
```

### Listing 30.66 Einrichten eines WireGuard-Tunnels auf der Konsole

Mit den Befehlen aus Listing 30.66 legen Sie zunächst die von WireGuard benötigten privaten und öffentlichen Schlüssel an. Anschließend wird die Schnittstelle `wg0` eingerichtet, mit dem privaten Schlüssel versorgt und gestartet. Im dritten Block wird das bisherige Ergebnis mit `wg` ausgegeben – dort erhalten Sie die wichtige Information, auf welchem Port der Tunnel läuft.

Das letzte Kommando startet den Tunnel mit den Daten des jeweiligen Partners. Anschließend können sich die Systeme `tom.example.com` und auch `jerry.example.com` wechselseitig über ihre internen IP-Adressen erreichen. Den Status des Tunnels können Sie sich einfach mit `wg show` anzeigen lassen. Listing 30.67 zeigt die Ausgabe des Tools:

```
interface: wg0
  public key: qMka[...]3R0=
  private key: (hidden)
  listening port: 34743

peer: F+Nk[...]imw=
  endpoint: 203.0.113.2:39478
  allowed ips: 10.0.0.2/32
  latest handshake: 1 hour, 42 minutes, 34 seconds ago
  transfer: 476 B received, 564 B sent
```

### Listing 30.67 Statusausgabe mit »wg show«

WireGuard ist im Übrigen ein sehr zurückhaltendes Protokoll. Werden keine Pakete über den Tunnel gesendet, gibt es für WireGuard auch nichts zu übertragen – dementsprechend sehen Sie nur minimalen Datenverkehr auf der Leitung, wenn gerade keine Daten gesendet



werden. Und genau so lange, wie keine Daten gesendet wurden, liegt auch der letzte *handshake* zurück (also das Aushandeln einer Verbindung) – im Beispiel aus Listing 30.67 also 1 Stunde, 42 Minuten und 34 Sekunden.

### 30.9.2 Die dunkle Seite des Mondes

An dieser Stelle sind alle, die bereits das Vergnügen hatten, einen IPsec-Tunnel aufzubauen, hellauf begeistert. Und das vollkommen zu Recht. Allerdings werden Ihnen sicherlich schon die Schattenseiten aufgefallen sein:

#### 1. Schlüsselaustausch

Mit WireGuard werden Punkt-zu-Punkt-Verbindungen hergestellt, denn es verfügt selbst über keine Mechanismen zum Schlüsselaustausch.

#### 2. Dynamische IP-Adressvergabe

Allen Teilnehmern muss vorab eine IP-Adresse zugewiesen werden. WireGuard selbst hat keine Mechanismen zur Vergabe von IP-Adressen.

#### 3. Weitere Verifizierungsmethoden

Für den Aufbau eines Tunnels genügt es, drei Variablen zu kennen: den Schlüssel, die interne und die externe IP-Adresse. Darüber hinaus werden keine weiteren Methoden umgesetzt.

#### 4. Kein Verbindungsaufbau über einen Web-Proxy-Server

Jede WireGuard-VPN-Verbindung hat eigene Ports und läuft ausschließlich über UDP.

### 30.9.3 Dauerhafte Tunnel mit »systemd«

Natürlich können Sie mit WireGuard nicht nur Tunnel zur Laufzeit erstellen, sondern diese auch permanent einrichten. Falls Sie unserem Beispiel bis hierher gefolgt sind, müssen Sie nicht bei null anfangen, um eine Konfiguration zu erstellen. Mit dem Befehl aus Listing 30.68 können Sie die bisherige Konfiguration speichern:

```
$ touch /etc/wireguard/wg0.conf
$ wg-quick save wg0
```

**Listing 30.68** Speichern des WireGuard-Tunnel »wg0«

Mit dem ersten Kommando wird die leere Datei *wg0.conf* angelegt, mit dem zweiten Kommando die zurzeit aktive Konfiguration dort hineingespeichert. In der Datei finden Sie nun die Zeilen aus Listing 30.69:

```
[Interface]
Address = 10.0.0.1/24
ListenPort = 34743
PrivateKey = +LVkL...Nw2I=
```

```
[Peer]
PublicKey = F+Nk...imw=
AllowedIPs = 10.0.0.2/32
Endpoint = 203.0.113.2:39478
```

**Listing 30.69** Inhalt der Konfigurationsdatei »wg0.conf«

Um diese Konfiguration, zum Beispiel nach einem Neustart, wieder zu aktivieren, genügt es, den Befehl aus Listing 30.70 abzusetzen:

```
$ wg-quick up wg0
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.0.0.1/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
```

**Listing 30.70** Den WireGuard-Tunnel »wg0« neu starten

Analog dazu können Sie den laufenden Tunnel (ob er per Konfiguration gestartet oder manuell eingerichtet wurde, spielt keine Rolle) mit dem Pendant `wg-quick down wg0` stoppen. Alle zurzeit verbundenen Tunnel können Sie sich ebenfalls mit dem Kommando `wg` anzeigen lassen (siehe Listing 30.71):

```
$ wg show all endpoints
wg0 F+Nk...jimw= 203.0.113.2:39478
   NCsP...OLzg= 192.0.2.17:57903
```

**Listing 30.71** Verbundene Tunnel anzeigen lassen

Wie Sie in Listing 30.71 sehen, ist es durchaus auch möglich, mehr als einen Tunnel mit einer Schnittstelle zu verbinden – ganz recht, auch *Hub and Spoke*-Netzwerke sind mit WireGuard möglich. Wie das genau funktioniert, zeigen wir Ihnen im nächsten Abschnitt. Nun aber zunächst zurück zum dauerhaften Tunnel. Dank der bereits gespeicherten Konfiguration und dem von WireGuard bereitgestellten `systemd`-Instanzskript (`wg-quick@.service`) können Sie ganz schnell einen dauerhaften Tunnel einrichten (siehe Listing 30.72):

```
$ systemctl start wg-quick@wg0.service
$ systemctl status wg-quick@wg0.service
* wg-quick@wg0.service - WireGuard via wg-quick(8) for wg0
   loaded: loaded (/lib/systemd/system/wg-quick@.service; disabled; vendor [...])
   Active: active (exited) since Sun 2023-01-08 14:57:52 UTC; 5s ago
     Docs: man:wg-quick(8)
           man:wg(8)
           https://www.wireguard.com/
           https://www.wireguard.com/quickstart/
           https://git.zx2c4.com/wireguard-tools/about/src/man/wg-quick.8
```

```

https://git.zx2c4.com/wireguard-tools/about/src/man/wg.8
Process: 3336 ExecStart=/usr/bin/wg-quick up wg0 (code=exited, status=0/SUCCESS)
Main PID: 3336 (code=exited, status=0/SUCCESS)
CPU: 44ms

```

```

Jan 08 14:57:52 ubuntu systemd[1]: Starting WireGuard via wg-quick(8) for wg0...
Jan 08 14:57:52 ubuntu wg-quick[3336]: [#] ip link add wg0 type wireguard
Jan 08 14:57:52 ubuntu wg-quick[3336]: [#] wg setconf wg0 /dev/fd/63
Jan 08 14:57:52 ubuntu wg-quick[3336]: [#] ip -4 address add 10.0.0.1/24 dev wg0
Jan 08 14:57:52 ubuntu wg-quick[3336]: [#] ip link set mtu 1420 up dev wg0
Jan 08 14:57:52 ubuntu systemd[1]: Finished WireGuard via wg-quick(8) for wg0.

```

#### Listing 30.72 systemd-Skript für »wg0« einrichten

Wie Sie in Listing 30.72 sehen, genügt es, dem systemd-Skript den Dateinamen (ohne Suffix `.conf`) als Parameter hinter dem At-Zeichen (`@`) zu übergeben. Anschließend nimmt WireGuard sofort seinen Dienst auf.

30

### 30.9.4 Alle machen mit: »Hub and Spoke«

Wie bereits angedeutet, können Sie mit WireGuard ebenfalls relativ einfach ein zentrales VPN-Netzwerk aufbauen, in dem sich alle Teilnehmer erreichen können. Dafür reicht es, wenn Sie ein System zur Zentrale machen – in unserem Beispiel verwenden wir dafür das System *tom.example.com*, da es die erste IP-Adresse des Netzwerksegments hat. Zusätzlich soll das System *spike.example.com* mit der IP-Adresse *10.0.0.3* (extern: *192.0.2.17*) hinzukommen. Fügen Sie der Konfigurationsdatei *wg0.conf* zunächst auf der Zentrale die Zeilen für einen neuen Teilnehmer (engl. *peer*) hinzu, so wie in Listing 30.73 dargestellt:

```

[Interface]
Address = 10.0.0.1/24
ListenPort = 34743
PrivateKey = +LVk...Nw2I=

[Peer]
PublicKey = F+NkM...jimw=
AllowedIPs = 10.0.0.2/32
Endpoint = 203.0.113.2:39478

[Peer]
PublicKey = NCsPm...OLzg=
AllowedIPs = 10.0.0.3/32
Endpoint = 192.0.2.17:57903

```

#### Listing 30.73 Zentrale: Inhalt der »wg0.conf«

Nach einem Neustart des Dienstes ist die Konfiguration auf der Zentrale bereits abgeschlossen. Auf den jeweiligen Endpunkten (*jerry.example.com* und *spike.example.com*) müssen Sie bei der Einrichtung des Tunnels vom bisherigen Pfad leicht abweichen. Alle Punkte sind identisch bis auf die *allowed-ips*. Diese müssen auf das gesamte Netzwerk erweitert werden und dürfen nicht wie bei der bisherigen Punkt-zu-Punkt-Verbindung auf dem System belassen werden. Ändern Sie daher den Wert auf `10.0.0.0/24`: Entweder über den Befehle `wg` (siehe Listing 30.66) oder in der gespeicherten Konfigurationsdatei – diese könnte dann so aussehen wie in Listing 30.74 dargestellt.

```
[Interface]
Address = 10.0.0.2/24
ListenPort = 39478
PrivateKey = cE70...UkUM=
```

```
[Peer]
PublicKey = qMKa...C3R0=
AllowedIPs = 10.0.0.0/24
Endpoint = 198.51.100.1:34743
```

**Listing 30.74** Außenstellen: Inhalt der »wg0.conf« (*jerry.example.com*)

### 30.9.5 Tipps und Tricks

In diesem Abschnitt wollen wir Ihnen noch ein paar Tipps und Tricks mit an die Hand geben, damit Sie dem einen oder anderen Fehler vielleicht schneller auf die Schliche kommen.

#### Verbindungsabbrüche

Wie bereits erwähnt, ist WireGuard ein sehr zurückhaltendes Protokoll. Es gibt nur etwas von sich, wenn es auch gefragt wurde. Was an und für sich eine gute Eigenschaft ist, kann auch zum Nachteil werden, sobald Firewalls ins Spiel kommen. So kann es nämlich vorkommen, dass Verbindungen abrechen, da die Sitzungen vermeintlich abgelaufen sind. Damit dies Ihrem Tunnel nicht passiert, können Sie zum Allheilmittel »keepalive« greifen. Fügen Sie dafür einfach den Parameter `PersistentKeepalive = <SEC>` hinzu, und ersetzen Sie den Platzhalter `<SEC>` durch die Anzahl der Sekunden – irgendwas unterhalb von 120 sollte passen, oft wird hier 25 empfohlen.

#### DNS-Namensauflösung

Wie so oft kann Ihnen auch bei WireGuard das DNS in die Suppe spucken. Wenn Sie den Parameter `DNS = <IP>` in einer Tunnelkonfiguration verwenden, dann wird WireGuard versuchen, die IP-Adresse als DNS-Server für die Schnittstelle einzurichten, wenn der Tunnel aufgebaut wird. Dabei setzt es `resolvconf` ein. Dieses Tool versteht sich mit der `systemd`-Alternative `systemd-resolve` nicht, wodurch Probleme entstehen können.

### Hostname anstelle von IP-Adressen

Falls Sie in Ihren Konfigurationsdateien für WireGuard Hostnamen anstelle von IP-Adressen verwendet haben, kann Ihnen das auf die Füße fallen. Sollte ein Name nicht auflösbar sein, so kann es beim Neustart dazu führen, dass Ihr System minutenlang an einer Stelle verharrt, bis es aufgibt. Nutzen Sie daher besser IP-Adressen, wo immer es möglich ist.

## 30.10 Fazit

WireGuard ist eine hervorragende Lösung, um schnell zwei (oder ein paar) Endpunkte miteinander zu verbinden. Sie haben damit eine einfache Möglichkeit, um mal schnell nach Hause zu funken. Und in der Tat ist WireGuard nicht nur schnell eingerichtet, sondern überträgt auch Ihre Daten extrem flott. Für große und komplexe Umgebungen sind andere Lösungen (wie z.B. OpenVPN) allerdings überlegen.



# Kapitel 31

## Verschlüsselung und Zertifikate

*Verschlüsselung und Zertifikate stellen elementare sicherheitsrelevante Themen dar. In diesem Kapitel bringen wir Ihnen diese Bereiche deshalb näher: von der eigenen Certificate Authority (CA) bis hin zu vollständig verschlüsselten Systemen.*

Das Bedürfnis, Nachrichten für niemanden – außer für den festgelegten Adressaten – lesbar zu übertragen, ist kein Phänomen der Neuzeit. Bereits im dritten Jahrtausend v. Chr. wurden in der altägyptischen Welt Nachrichten verschlüsselt. Selbstverständlich wurden die Methoden über die Jahrhunderte hinweg stets erweitert und verbessert. In diesem Kapitel lernen Sie die Funktionsweise von Verschlüsselungen kennen sowie deren Entwicklung und die Formen der Umsetzung (durch Zertifikate) in der heutigen Zeit.

31

### 31.1 Definition und Historie

Die Wissenschaft der *Kryptologie* unterteilt sich in zwei große Bereiche. Auf der einen Seite steht die *Kryptografie*, die sich der Entwicklung und Anwendung der einzelnen Verfahren widmet. Auf der anderen Seite steht die *Kryptoanalyse*. Diese befasst sich mit kryptografischen Verfahren (nicht nur zur Verschlüsselung), um entweder deren Schutzfunktion aufzuheben bzw. zu umgehen, um diese somit zu »entziffern«, oder um ihre Sicherheit nachzuweisen und zu quantifizieren. Sie bildet somit den »Gegenspieler« der Kryptografie.

Die Kryptografie beschreibt allgemein die Umformung von lesbarem Text, dem sogenannten *Klartext*, in einen vermeintlich unlesbaren Text, den *Geheimtext*. In den Anfängen der Verschlüsselungstechnik wurden dafür einfachste Methoden angewandt.

Bereits 3000 v. Chr. wurde beispielsweise die *Atbasch*-Verschlüsselung angewandt. Bei dieser Art der Verschlüsselung wird der Abstand des Buchstabens zum Beginn des Alphabets ermittelt. Dem entsprechenden Buchstaben wird dann der Buchstabe zugeordnet, der den gleichen Abstand vom Ende des Alphabets her hat.

A --> Z  
B --> Y  
C --> X  
...

**Listing 31.1** Atbasch-Verschlüsselung

Eine weitere sehr bekannte Verschlüsselung ist die zu Zeiten Caesars nach ihm benannte *Caesar-Verschlüsselung*. Sie stellt eine einfache *Verschiebe-Chiffre* dar. Hierbei werden die Buchstaben des Alphabets um einen definierten Faktor verschoben (siehe Abbildung 31.1).

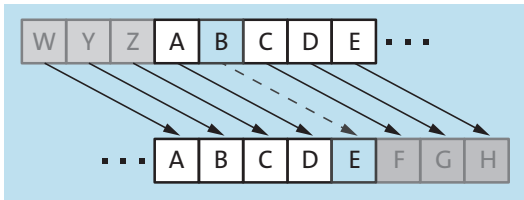


Abbildung 31.1 Caesar-Verschlüsselung mit dem Faktor 3

So ergibt der folgende Satz ...

Hallo Welt ich bin wieder da

Listing 31.2 Caesar-Verschlüsselung: Klartext

... nach einer Verschiebung um drei Stellen, was in der Caesar-Verschlüsselung eine *c-Kodierung* genannt wird, folgenden unleserlichen Geheimtext:

KDOOR ZHOW LFK ELQ ZLGHU GD

Listing 31.3 Caesar-Verschlüsselung: Geheimtext

Auf dieses Verfahren setzte das *Vigenère-Verfahren* auf. Bei diesem im 16. Jahrhundert von Blaise de Vigenère entwickelten Verfahren wird die Caesar-Verschlüsselung mehrfach angewandt. Es handelt sich bei dieser Verschlüsselungsform um die erste bekannte Verschlüsselung, die einen »echten« Schlüssel einsetzt. Das Verfahren konnte erst im Jahre 1854 von Charles Babbage geknackt werden. Bei dem Verfahren wird der Klartext mithilfe des Schlüssels um den Faktor der Position des jeweiligen Buchstabens des Schlüssels im Alphabet verschoben. Der Schlüssel wird dabei entsprechend der Länge des Klartextes wiederholt. Tabelle 31.1 zeigt ein Beispiel.

| Attribut      | Wert                            |
|---------------|---------------------------------|
| Klartext      | DiesIstDerGeheimText            |
| Schlüsselwort | anykeyanykeyanykeyan            |
| Abstände      | a=0; n=12; y=24; k=9; e=4; y=24 |
| Geheimtext    | DvccMqtQcbKchrgwXcxg            |

Tabelle 31.1 Vigenère-Verschlüsselung



Die Einsatzgebiete waren meist politischer/militärischer Natur. Der hauptsächliche Sicherheitsgewinn beruhte darauf, dass möglichst niemand außer den Beteiligten das Verfahren kannte. Diese Vorgehensweise wird auch als *Security by obscurity* bezeichnet.

Nach Vigenère wurden Verschlüsselungen mehr und mehr mathematisch und basierten fast ausschließlich auf Schlüsseln. Als Wendepunkt der Kryptografie hin zur heutigen Wissenschaft gelten die im Jahre 1883 von *Auguste Kerckhoff* formulierten Grundsätze:

1. *Das System muss im Wesentlichen (...) unentzifferbar sein.*
2. *Das System darf keine Geheimhaltung erfordern (...).*
3. *Es muss leicht übermittelbar sein, und man muss sich die Schlüssel ohne schriftliche Aufzeichnung merken können (...).*
4. *Das System sollte mit telegrafischer Kommunikation kompatibel sein.*
5. *Das System muss transportabel sein, und die Bedienung darf nicht mehr als eine Person erfordern.*
6. *Das System muss einfach anwendbar sein (...).*

Diese Grundsätze sorgten dafür, dass das bis dahin vorherrschende Prinzip der *Security by obscurity* an Bedeutung verlor. Kerckhoffs Grundsätze verdeutlichen, dass eine größere Sicherheit gewonnen werden kann, wenn der verwendete Algorithmus bekannt und somit prüfbar und verifizierbar ist.

## 31.2 Moderne Kryptologie

Die heutige Wissenschaft der Kryptologie unterscheidet zwei wesentliche Konzepte zur Verschlüsselung. Dabei handelt es sich zum einen um das *symmetrische Verschlüsselungsverfahren* und zum anderen um das *asymmetrische Verschlüsselungsverfahren*.

### 31.2.1 Symmetrische Verschlüsselung

Bei der symmetrischen Verschlüsselung besitzen beide Parteien den gleichen Schlüssel, um den Geheimtext zu dechiffrieren, den sogenannten *PSK*<sup>1</sup>.

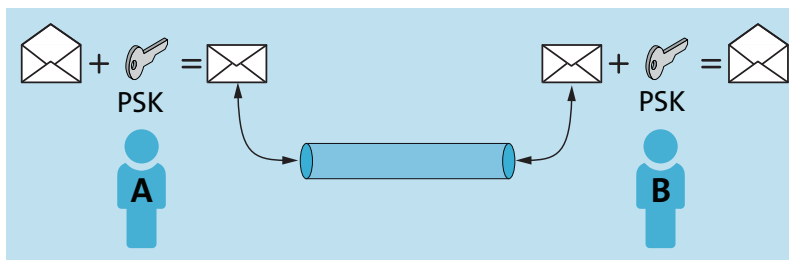
Dies wirft allerdings das Problem auf, wie der PSK »sicher« übertragen werden kann, damit die Nachricht auch nur von den beteiligten Parteien gelesen werden kann. Die Verschlüsselungsverfahren, die eine symmetrische Verschlüsselung einsetzen, haben wir für Sie in Tabelle 31.2 zusammengefasst. Abbildung 31.2 verdeutlicht die Funktionsweise eines symmetrischen Verschlüsselungsverfahrens.

---

<sup>1</sup> *Pre-Shared Key*, engl. für *vorher ausgetauschter Schlüssel*

| Bezeichnung     | Veröff. | Erläuterung                                             |
|-----------------|---------|---------------------------------------------------------|
| Lucifer         | 1973    | von Horst Feistel (IBM), Blockchiffre von 128 Bit       |
| DES             | 1977    | <i>Data Encryption Standard</i> (56 Bit)                |
| 3DES            | 1999    | dreifach angewandeter DES (168 Bit)                     |
| AES             | 2001    | <i>Advanced Encryption Standard</i> (128, 192, 256 Bit) |
| ChaCha_Poly1305 | 2008    | neuer Cipher von Bernstein/Lange                        |

**Tabelle 31.2** Liste der symmetrischen Verschlüsselungsverfahren



**Abbildung 31.2** Ablauf eines symmetrischen Verschlüsselungsverfahrens



Person A und Person B verschlüsseln jeweils ihre Nachrichten mit dem PSK. Da nur sie im Besitz des PSK sind, können auch nur sie die Nachrichten entschlüsseln. Entscheidend für symmetrische Verschlüsselungsverfahren ist die Schlüssellänge. Ist diese zu kurz, können moderne Computer mit der *Brute-Force-Methode*<sup>2</sup>, die auch als *Exhaustionsmethode*<sup>3</sup> bezeichnet wird, überwunden werden. Dabei werden alle erdenklichen Schlüssel ausprobiert, bis der richtige gefunden ist. Mit entsprechenden mathematischen Verfahren können auf kurzen Schlüssellängen beruhende Verfahren wie DES in adäquater Zeit gebrochen werden.

### 31.2.2 Asymmetrische Verschlüsselung

Bei der asymmetrischen Verschlüsselung existiert ein Schlüsselpaar: jeweils ein privater (*Private Key*) und ein öffentlicher Schlüssel (*Public Key*). Nachrichten werden jeweils mit dem öffentlichen Schlüssel des Kommunikationspartners verschlüsselt und können nur mit dessen privatem Schlüssel entschlüsselt werden. Dieses Verfahren wird als *Public-Key-Verfahren* bezeichnet und basiert auf Einwegfunktionen, wie den Hash-Funktionen *md5* oder *sha-1*.

<sup>2</sup> *Brute Force*, engl. für *rohe Gewalt*

<sup>3</sup> *Exhaustionsmethode*, von lat. *exaurire* = ausschöpfen

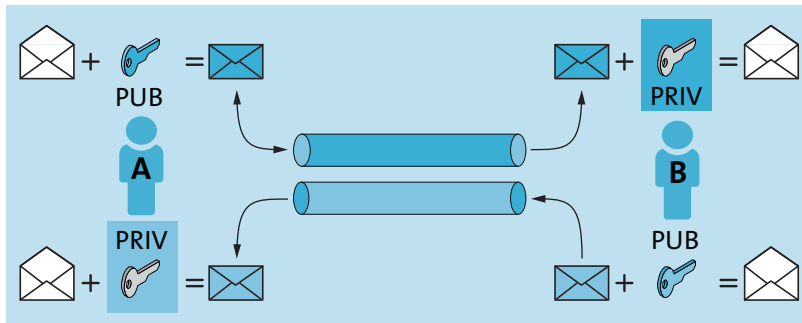


Abbildung 31.3 Ablauf eines asymmetrischen Verschlüsselungsverfahrens

Abbildung 31.3 verdeutlicht das Prinzip: *Person A* verschlüsselt eine Nachricht für *Person B* mit deren öffentlichem Schlüssel (PUB). *Person B* kann die Nachricht mit ihrem privaten Schlüssel (PRIV) entschlüsseln. Die Antwort von *Person B* wird mit dem öffentlichen Schlüssel von *Person A* verschlüsselt. Diese Nachricht kann wiederum nur von *Person A* mit ihrem privaten Schlüssel entschlüsselt werden. Heutzutage werden überwiegend semi-asymmetrische Verfahren eingesetzt. Dabei wird das asymmetrische Verfahren aufgrund des hohen Rechenaufwands lediglich auf die Schlüssel der mit symmetrischen Verfahren verschlüsselten Daten angewandt.



## 31.3 Den Durchblick behalten

In der IT werden Verschlüsselungen hauptsächlich zur Absicherung von Kommunikation, zur sicheren Aufbewahrung von Daten und zur Prüfung der Integrität und Authentizität verwendet. Dabei entsteht das Problem, die benötigten Schlüssel sicher zu übertragen.

### 31.3.1 Das Grundproblem

Jede Verschlüsselung muss folgende drei Punkte sicherstellen:

- ▶ **Vertraulichkeit** – Nur dazu berechtigte Personen sollen in der Lage sein, Daten zu entschlüsseln.
- ▶ **Integrität** – Die Daten müssen vollständig und unverändert sein.
- ▶ **Authentizität** – Der Absender muss identifizierbar sein.

Zwei Konzepte haben sich zum Erreichen dieser Ziele durchgesetzt: zum einen die *Public Key Infrastructure (PKI)*<sup>4</sup> und zum anderen das Prinzip der *Pretty Good Privacy (PGP)*.

<sup>4</sup> *Public Key Infrastructure*, engl. für *Zertifikatsinfrastruktur*

### 31.3.2 Verwendungszwecke

Vorab muss noch darauf hingewiesen werden, dass sich Verschlüsselung nicht nur auf Daten oder Nachrichten bezieht. Typische Anwendungen von Verschlüsselungen sind:

- ▶ **elektronische Signaturen** (elektronischer Ausweis)
- ▶ **Absicherung von Netzwerkprotokollen** (zum Beispiel HTTPS, IPsec oder SSH)
- ▶ **Schutz von E-Mails** (S/MIME oder PGP)

### 31.3.3 Umsetzung mithilfe einer PKI

Eine *Public Key Infrastructure* stellt eine hierarchische Struktur bereit. Eine Kopfstelle (*CA*<sup>5</sup>) stellt Zertifikate für Personen oder Server aus. Diese Zertifikate enthalten alle notwendigen Daten und können über die CA geprüft werden. Eine PKI besteht aus den in Abbildung 31.4 dargestellten Elementen.

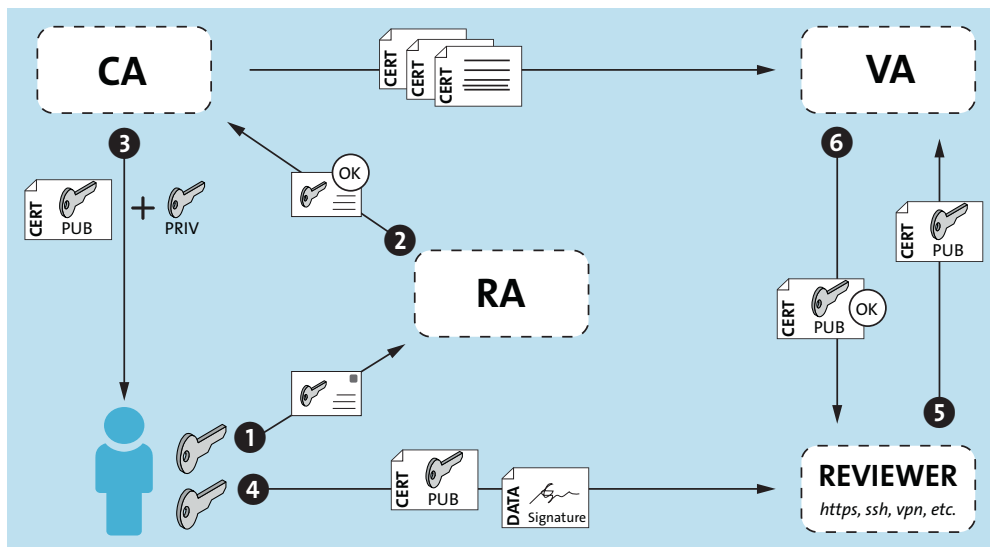


Abbildung 31.4 Aufbau einer PKI

Der Antragsteller identifiziert sich, je nach benötigtem »Grad« der Sicherheit, bei der *RA*<sup>6</sup>, Schritt (1). Dies kann über die Angabe einer E-Mail-Adresse erfolgen, über die Postzustellung einer Kopie des Ausweises oder über weitere Maßnahmen. Die RA prüft die Daten und erteilt der CA, wenn die entsprechenden Anforderungen erfüllt sind, den Auftrag, ein Zertifikat für den Antragsteller zu erstellen (2). Dieses Zertifikat (3) besteht wiederum aus zwei Elementen:

5 *Certificate Authority*, engl. für *Zertifizierungsstelle*

6 *Registration Authority*, engl. für *Registrierungsstelle*

zum einen aus dem eigentlichen Zertifikat, an das der öffentliche Schlüssel angehängt ist, und zum anderen aus dem privaten Schlüssel. Die CA teilt die ausgestellten Zertifikate wiederum der VA<sup>7</sup> mit. Über die VA kann die Gültigkeit des Zertifikats von Dritten (*Reviewer* in Abbildung 31.4) geprüft werden (4–6). Ein weiterer wesentlicher Bestandteil einer PKI ist die CRL<sup>8</sup>. Wenn ein Zertifikat korrumpiert wurde oder verloren ging, kann es gesperrt werden. Die zuständige CA nimmt dieses Zertifikat dann in die CRL auf, sodass bei einer Prüfung das Zertifikat abgelehnt wird. Die Prüfung kann über den Download von CRLs erfolgen oder aber online. Dafür wurde das *Online Certificate Status Protocol* (OCSP) geschaffen, das eine Online-Abfrage ermöglicht. Es wird zwischen privaten und offiziellen PKIs unterschieden. Der Verschlüsselungsstandard einer privaten PKI entspricht dem einer öffentlichen, nur dass bei einer privaten PKI nicht für jedermann die Gültigkeit und Authentizität geprüft werden kann. Gerade im Bereich Webserver (HTTPS) empfiehlt es sich daher, ein offizielles Zertifikat einzusetzen. Diese kostenpflichtigen offiziellen Zertifikate verhindern, dass die Benutzer eine Warnmeldung erhalten, da ein selbst erstelltes Zertifikat nicht bei einer der bekannten Stellen geprüft werden kann. Diese offiziellen PKIs werden als *TrustCenter* bezeichnet. In Deutschland gibt es eine Vielzahl von TrustCentern; hier eine kleine Auswahl:

- ▶ D-TRUST ([www.d-trust.net](http://www.d-trust.net))
- ▶ Telesec ([www.telesec.de](http://www.telesec.de))
- ▶ Verisign ([www.verisign.de](http://www.verisign.de))
- ▶ PSW Group ([www.psw.net](http://www.psw.net))

#### 31.3.4 X.509

Bei X.509 handelt es sich um den De-facto-Standard für Zertifikate. Ein Zertifikat, das nach der aktuellen Version X.509v3 erstellt wurde, beinhaltet folgende Elemente:

- ▶ Version
- ▶ Seriennummer
- ▶ Algorithmen-ID
- ▶ Aussteller
- ▶ Gültigkeit (von/bis)
- ▶ Zertifikatsinhaber
- ▶ Subject Public Key Info:
  - Public-Key-Algorithmus
  - Subject Public Key

<sup>7</sup> *Validation Authority*, engl. für *Validierungsdienst*

<sup>8</sup> *Certificate Revocation List*, engl. für *Zertifikatssperrliste*

- ▶ eindeutige ID des Ausstellers (optional)
- ▶ eindeutige ID des Inhabers (optional)
- ▶ Erweiterungen
- ▶ Zertifikatssignaturalgorithmus
- ▶ Zertifikatssignatur

Diese Zertifikate können nicht nur für Personen ausgestellt werden, sondern ebenso für Gruppen, Server, Clients oder explizite andere Verwendungszwecke. X.509-Zertifikate werden für Einzelpersonen in vier unterschiedlichen Klassen ausgestellt:

- ▶ **Class 0**  
Demo-Zertifikate, die nur für Testzwecke gedacht und mit beschränkter Gültigkeitsdauer versehen sind
- ▶ **Class 1**  
Zertifikate mit geringem Nachweis der Identität. Es wird lediglich sichergestellt, dass eine E-Mail-Adresse existiert und der Besitzer des öffentlichen Schlüssels Zugriff auf dieses E-Mail-Konto besitzt. Diese Zertifikate finden oft im privaten Sektor Anwendung.
- ▶ **Class 2**  
Zertifikate dieser Klasse werden auch als *Unternehmenszertifikate* bezeichnet. Hierbei genügen eine Kopie des Handelsregistrauszugs zur Feststellung der zeichnungsberechtigten Person sowie ein schriftlicher Auftrag.
- ▶ **Class 3**  
Zertifikate mit persönlicher Identitätsprüfung. Der Aussteller bestätigt mit diesem Zertifikat, dass der Besitzer einwandfrei identifiziert wurde (Personalausweis/Reisepass). Hauptsächlich Einsatzgebiete sind das Internet-Banking, das Online-Shopping und die elektronische Kommunikation mit Behörden (Übermittlung von Steuerdaten und die elektronische Antragstellung).

Zertifikate werden in unterschiedlichen Formaten ausgeliefert. Tabelle 31.3 zeigt übliche Formate (Dateinamenerweiterungen) für X.509-Zertifikate.

| Endung | Typ                                                                                                                                                                                                            |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| .CER   | <i>Canonical Encoding Rules</i> -kodierte Zertifikat oder Zertifikatsfolgen                                                                                                                                    |
| .CRT   | <i>Distinguished Encoding Rules</i> - oder <i>Base64</i> -kodierte Zertifikat                                                                                                                                  |
| .CSR   | Base64-kodierte Zertifizierungsanfrage des öffentlichen Schlüssels (plus weitere Metadaten des Besitzers) an eine CA, umschlossen von -----BEGIN CERTIFICATE REQUEST----- und -----ENDCERTIFICATE REQUEST----- |

**Tabelle 31.3** Dateinamenerweiterungen für X.509-Zertifikate

| Endung | Typ                                                                                                    |
|--------|--------------------------------------------------------------------------------------------------------|
| .DER   | <i>Distinguished Encoding Rules</i> -kodiertes Zertifikat                                              |
| .P12   | PKCS#12, kann öffentliche Zertifikate und private Schlüssel (kennwortgeschützt) enthalten.             |
| .P7C   | PKCS#7-signierte Datenstruktur ohne Dateninhalt, nur mit Zertifikat(en) oder Zertifikatsperrliste(n)   |
| .P7B   | siehe .P7C                                                                                             |
| .PEM   | Base64-kodiertes Zertifikat, umschlossen von -----BEGIN CERTIFICATE----- und -----END CERTIFICATE----- |
| .PFX   | siehe .P12                                                                                             |

Tabelle 31.3 Dateinamenerweiterungen für X.509-Zertifikate (Forts.)

### 31.3.5 Ein anderer Ansatz: PGP (Web-of-Trust)

Bei einem *Web-of-Trust* handelt es sich um eine sogenannte *Vertrauenskette*. Dieser liegt folgende Überlegung zugrunde:

*Person Alpha vertraut der Person Beta. Da Person Beta auch Person Gamma vertraut, kann auch Person Alpha der Person Gamma vertrauen.*

Dieses Prinzip findet bei *Pretty Good Privacy (PGP)* Anwendung. Dieses von Peter Zimmermann initiierte Konzept wurde geschaffen, um jedermann das Recht auf Privatsphäre auch im E-Mail-Verkehr zu ermöglichen. Da dieses Tool von den US-Behörden als starke Kryptografie eingestuft wurde und somit unter das Waffenexportgesetz fiel, konnte die Veröffentlichung in Europa nur über einen Trick erfolgen: Peter Zimmermann veröffentlichte den Quellcode in Buchform, sodass dieser in Europa nachgebildet werden konnte.

Da PGP patentgeschützt ist, gibt es die freie Implementierung *GnuPG (GNU Privacy Guard)*, die nach demselben Verfahren arbeitet, dabei aber ausschließlich auf patentfreie Algorithmen zurückgreift. Mit diesem Verfahren können Daten/Nachrichten signiert und sogar vollständig verschlüsselt werden. GnuPG implementiert sowohl den OpenPGP-Standard als auch – seit der Version 2.0 – S/MIME.

## 31.4 Einmal mit allem und kostenlos bitte: Let's Encrypt

Seit Ende 2015 wird das Feld der Zertifikate von einem neuen Spieler ordentlich aufgemischt. Die Rede ist natürlich von *Let's Encrypt*. Let's Encrypt (zu Deutsch *Lasst uns verschlüsseln*)

ist eine Zertifizierungsstelle, die kostenlose X.509-Zertifikate für *Transport Layer Security (TLS)* anbietet. Dies wird durch einen automatisierten Prozess erreicht, der die komplexen manuellen Vorgänge bei der Erstellung übernimmt, wie die Validierung, die Signierung, die Einrichtung und die Erneuerung von Zertifikaten. Das Projekt verfolgt das Ziel, verschlüsselte Verbindungen im Internet zum Standard zu machen.

Dazu betreibt das Projekt eine Ausgabestelle für Zertifikate. Um es uns Administratoren möglichst leicht zu machen, bietet das Projekt einen Software-Client, der sich um das Erstellen, Signieren, Validieren und das rechtzeitige Erneuern der Zertifikate kümmert. Der originäre Client des Projekts heißt *certbot*. Auf der Projektseite unter <https://certbot.eff.org/> finden Sie viele Hinweise und eine umfangreiche Dokumentation. Selbstverständlich sind die Root-CA-Zertifikate von Let's Encrypt in allen gängigen Browsern enthalten, sodass es zu keinerlei Warnmeldungen kommt.

### 31.4.1 Wie funktioniert das?

Das Prinzip ist relativ simpel. Wie alle Zertifizierungsstellen muss auch Let's Encrypt sicherstellen, dass ein Zertifikat nur an berechtigte Personen ausgestellt wird. Anstatt dies aber kompliziert über Ident-Verfahren zu lösen, haben die Entwickler eine automatisierte Variante erdacht:

1. Der Let's-Encrypt-Client fragt zunächst die CA, was zu tun ist, um zu beweisen, dass die Domäne, für die ein Zertifikat ausgestellt werden soll, auch unter der Kontrolle des Beantragenden ist, und überträgt den öffentlichen Teil des Zertifikats.
2. Die Antwort der CA besteht in drei Möglichkeiten, den Beweis anzutreten:
  - HTTP-01: Einrichtung einer *HTTP Resource* für die Domäne (z. B. <http://example.com/4711>)
  - DNS-01: Einrichtung eines angepassten *DNS Resource Records* für die Domäne
  - TLS-ALPN-01: Einrichtung einer *HTTP Resource* via TLS mit SNI (Dieses Verfahren setzte das ALPN-Protokoll voraus und wird nur von wenigen Webservern und Clients unterstützt.)

In der Regel wird die HTTP-01-Methode verwendet. Gleichzeitig wird ein *nonce*<sup>9</sup> übertragen (zum Beispiel *abcd*), das vom Client mit seinem privaten Schlüssel signiert werden muss.

3. Der Client erfüllt nun eine der beiden Anforderungen. Meist nutzt der die Methode HTTP-01: Er legt also die angeforderte Datei (*4711*) an und legt dort den mit seinem privaten Schlüssel signierten *nonce* (*abcd*) ab.

---

<sup>9</sup> *nonce* bezeichnet in der Kryptografie einen vorläufigen Platzhalter, der zeitnah ersetzt werden sollte.



4. Die CA prüft nun, ob alles mit rechten Dingen zugeht – sprich, ob der *nonce* (*abcd*) unter <http://example.com/4711> erreichbar ist und korrekt signiert wurde. Wenn ja, gibt die CA Rückmeldung an den Client und stellt das Zertifikat aus.
5. Der Client erhält nun das von der CA signierte Zertifikat, und der Prozess ist abgeschlossen.

Die Kommunikation erfolgt im Übrigen über das Let's-Encrypt-eigene Protokoll *Automatic Certificate Management Environment* kurz *ACME* – und ja, die Anspielung auf die fiktive Firma der *Road Runner*-Cartoons ist natürlich reiner Zufall.

### 31.4.2 Einschränkungen

Anders als von sonstigen TrustCentern werden Lets-Encrypt-Zertifikate mit einer Laufzeit von »nur« 90 Tagen ausgegeben. Dies dient vor allem der Transparenz und dazu, aktuellen Veränderungen zeitnah Rechnung tragen zu können. Da Let's Encrypt mit einer Automation arbeitet, wurden ein paar Einschränkungen eingebaut, um Missbrauch zu verhindern. Diese sind aber so großzügig bemessen, dass nichts gegen einen produktiven Einsatz spricht:

- ▶ **Namen pro Zertifikat**  
sind zurzeit auf 100 Namen (oder Webseiten) pro Zertifikat beschränkt.
- ▶ **Zertifikate pro Domäne**  
sind auf 50 pro Woche limitiert. Dabei kann eine Anfrage eine oder mehrere Domänen beinhalten. Ausnahme: Wenn ein Zertifikat mit den gleichen Domänen abgefragt wird, greift diese Beschränkung nicht – aber gegebenenfalls die folgende Beschränkung.
- ▶ **Zertifikate pro FQDN-Set**  
beschränkt die Anzahl der Zertifikate, die mit der exakt gleichen Anzahl von FQDNs (die als *FQDN-Set* bezeichnet wird) abgerufen werden dürfen, auf 5 Zertifikate pro Woche.
- ▶ **Registrierung von IP-Adressen**  
ist derzeit auf 500 IP-Adressen pro 3 Stunden limitiert.
- ▶ **Ausstehende Autorisierung pro Account**  
begrenzt die Anzahl an Autorisierungsversuchen, ohne dass diese erfolgreich beendet werden, auf 300 pro Account pro Woche.

Im Regelbetrieb werden diese Werte nicht erreicht werden, auch dann nicht, wenn Sie viele Domänen über ein System verwalten (zum Beispiel auf einem Reverse-Proxy).

### 31.4.3 Der Client certbot

Der *certbot* wird über die Projektseite [certbot.eff.org](https://certbot.eff.org) und GitHub angeboten und besteht im Wesentlichen aus Python-Skripten. Diese sind praktischerweise so konzipiert, dass sie auf allen gängigen Distributionen lauffähig sind und sogar als Paket angeboten werden. In-

stallieren Sie daher einfach das Paket *certbot* aus den Paketquellen (bei CentOS müssen Sie vorher noch die EPEL-Quellen mittels `dnf install -y epel-release` aktivieren).

Um ein Zertifikat für die Domain *example.com* zu erhalten, genügt das Kommando aus Listing 31.4. Das Kommando leitet Sie durch die notwendigen Schritte und legt für Sie einen Account für die angegebene E-Mail-Adresse an (im Beispiel *daniel@example.com*).

```
root@ubuntu:~# certbot certonly --standalone --agree-tos -m daniel@example.com \
-d example.com
```

**Listing 31.4** Ein Zertifikat für »*example.com*« erstellen

`certonly` bedeutet, dass die Zertifikate nach dem Erzeugen nur im Filesystem abgelegt werden. Für bestimmte Dienste, etwa den Apache Webserver, kann `certbot` Ihnen auch etwas Arbeit abnehmen und die Pfade zu den Zertifikaten in die Webserver-Konfiguration einbinden und sogar die Validierung der Domäne abwickeln. Mit dem Parameter `--standalone` weisen Sie den `certbot` hingegen an, selbstständig einen Webserver zu starten.

Mit dem Parameter `--agree-tos` stimmen Sie den Nutzungsbedingungen automatisch zu. An die mit dem Parameter `-m` angegebene E-Mail-Adresse werden im Übrigen Ablauf- und Sicherheitsmeldungen versandt. Zu guter Letzt werden mit dem Parameter `-d` Domänennamen angegeben, für die Sie ein Zertifikat ausgestellt bekommen möchten – durch die mehrfache Angabe des Parameters können Sie ein Zertifikat anfordern, das für mehrere Domänen gültig ist. Diese werden dann als *Subject Alternative Name (SAN)* im Zertifikat angegeben.

Standardmäßig wird nun das HTTP-Verfahren zur Domänenprüfung angewandt. Ist es erfolgreich (wie im Beispiel aus Listing 31.4), dann werden die Zertifikate im Verzeichnis `/etc/letsencrypt/live/` abgelegt. Dort wird ein Verzeichnis angelegt, das nach dem zuerst angegebenen Domänennamen benannt wird. Dort finden Sie die Zertifikatsdateien (siehe Listing 31.5):

```
root@ubuntu:~# ls -l /etc/letsencrypt/live/<DOMAIN>/
cert.pem
chain.pem
fullchain.pem
privkey.pem
README
```

**Listing 31.5** Zertifikatsdateien unter »`/etc/letsencrypt/live/`«

Wie Sie in Listing 31.5 sehen, lagern dort nicht nur das klassische Zertifikat (*cert.pem*) mit dem dazugehörigen Schlüssel (*privkey.pem*), sondern auch direkt die Zertifikatskette der CAs (*chain.pem*) und die gesamte Zertifikatskette der CAs inklusive ihrem Serverzertifikat (*fullchain.pem*). In der Datei *README* finden Sie im Übrigen eine Zusammenfassung der Dateien und erfahren, wie Sie mit ihnen umgehen sollten.

Nach dem ersten Aufruf des *certbot* müssen Sie Ihre Zustimmung zu den Nutzungsbedingungen und Ihre E-Mail-Adresse nicht erneut angeben. Das Tool speichert Ihren Account und nutzt diesen automatisch, wenn Sie keinen anderen angeben. Im Übrigen können Sie fast alle Aktionen des *certbot* mit dem Parameter `--dry-run` testen, ohne Gefahr zu laufen, etwas »kaputt« zu machen. Wie der Name des Parameters bereits vermuten lässt, sorgt er für einen Trockendurchlauf, ohne »wirklich« etwas zu tun.

Je nach Webserver, den Sie einsetzen, können Sie nun die benötigten Varianten der Zertifikate an ihren Bestimmungsort bringen. Falls Ihnen das zu mühsam ist, können Sie dem Befehl aus Listing 31.4 noch den Parameter `--webroot <PFAD>` hinzufügen. Die Zertifikate werden dann von *certbot* direkt dort abgelegt. Im Übrigen sind für die gängigsten Webserver auch schon Pfade vordefiniert, sodass Sie *certbot* auch direkt darauf ansetzen können, die Pfade selbst aus Ihrer Konfiguration zu ermitteln und die Zertifikate entsprechend einzusortieren. Allerdings empfehlen wir Ihnen, dies nicht zu tun – falls ein Automatismus an dieser Stelle sinnvoll ist, sollten Sie ihn lieber selbst in ein bash-Skript umsetzen, so haben Sie die volle Kontrolle. Wie bereits erwähnt wurde, haben die Zertifikate von Let's Encrypt nur eine Gültigkeit von 90 Tagen. Daher müssen Sie den Befehl aus Listing 31.4 auch entsprechend häufig ausführen und die neuen Zertifikate direkt an Ihren Webserver (oder dergleichen) weitergeben. Dafür wurde bei der Installation unter `/etc/cron.d` direkt ein entsprechender Job angelegt – dieser läuft zyklisch und aktualisiert alle gefundenen Zertifikate automatisch.

## 31.5 In der Praxis

In diesem Abschnitt erfahren Sie alles, was Sie zum Betrieb oder zur Anwendung der beiden Konzepte PKI und Web-of-Trust benötigen, und Sie lernen, wie Sie diese am besten einsetzen.

### 31.5.1 Einrichtung einer PKI mit Server- und E-Mail-Zertifikaten

Eine eigene PKI zu betreiben kann durchaus Vorteile haben: nicht nur für das firmeninterne VPN, sondern auch, wenn Sie zusätzlich interne Webserver absichern wollen. Durch den hierarchischen Aufbau einer PKI können Sie für jedes Umfeld Unter-CAs bilden, die explizit für den jeweiligen Verwendungszweck vorgesehen sind – diese werden auch als *Intermediate-CAs* bezeichnet. Dies birgt nicht nur organisatorische Vorteile in sich, einige Anwendungen setzen dies sogar voraus. Seit der Version 3 sind in X.509-Zertifikaten erweiterte Attribute vorhanden, wie zum Beispiel *Key Usage* oder *Extended Key Usage*. Diese werden teilweise zwingend vorausgesetzt: Ohne diese verweigern einige Anwendungen den Dienst. Der Weg zur eigenen PKI besteht aus fünf Schritten:

1. Verzeichnisstruktur erstellen
2. OpenSSL-Konfiguration

3. CA erzeugen
4. Unter-CAs erzeugen
5. Zertifikate ausstellen

Anschließend können Sie selbst signierte Zertifikate für die entsprechenden Verwendungszwecke erstellen. Selbstverständlich können Sie die Struktur auch mit weiteren Unter-CAs erweitern. Hierfür kommt *OpenSSL* zum Einsatz, eine freie Implementierung der *SSL/TLS*-Protokolle.

Es bietet weitergehende Funktionen zur Zertifikatserstellung und -verwaltung sowie unterschiedliche kryptografische Funktionen. Das von Eric A. Young und Tim Hudson entwickelte *SSLey* diente dabei als Basis. Derzeit wird es von einer unabhängigen Gruppe weiterentwickelt. Es ist Bestandteil fast jeder Linux-Distribution. Auch das bekannte SSH setzt zur Verschlüsselung *OpenSSL* ein.

### Verzeichnisstruktur erstellen

Erstellen Sie zunächst die folgende Verzeichnisstruktur in einem beliebigen Unterverzeichnis. Im Beispiel wurde das Verzeichnis *CA* im Homeverzeichnis des *root*-Benutzers gewählt.

```
.
|-- RootCA
|   |-- certs
|   |-- crl
|   |-- newcerts
|   |-- private
|   |--index.txt
|   `--serial
|-- ServerCA
|   |-- certs
|   |-- crl
|   |-- newcerts
|   |-- private
|   |--index.txt
|   `--serial
`-- EMailCA
    |-- certs
    |-- crl
    |-- newcerts
    |-- private
    |--index.txt
    `--serial
```

**Listing 31.6** CA-Verzeichnisstruktur unterhalb von »/root/CA«

Mit den folgenden Zeilen können Sie alle Verzeichnisse und Dateien auf einmal erstellen:

```
ca:~/CA # mkdir -p CA/{RootCA,ServerCA,EMailCA}/{certs,crl,newcerts,private}
ca:~/CA # touch CA/{RootCA,ServerCA,EMailCA}/index.txt
ca:~/CA # echo "00" > CA/{RootCA,ServerCA,EMailCA}/serial
```

**Listing 31.7** Anlegen der benötigten Verzeichnisse und Dateien

Bei der *RootCA* handelt es sich um die Stammzertifizierungsstelle. Diese stellt das Root-Zertifikat aus, also das oberste Zertifikat der Vertrauenskette. Des Weiteren erzeugt die *RootCA* Zertifikate für die Unter-CAs (*ServerCA* und *EMailCA*). Fügen Sie das *RootCA*-Zertifikat (öffentlicher Schlüssel) einem System hinzu, akzeptiert dieses System aufgrund der hierarchischen Struktur alle Zertifikate, die von jedweder Unter-CA erstellt wurden, solange ihm die Kette mit übergeben wird (ein häufig vorkommender Fehler bei Webserverkonfigurationen).

Bei *ServerCA* und *EMailCA* handelt es sich um solche Unter-CAs. Diese werden ihren Namen entsprechend eingesetzt: zum einen, um Serverzertifikate, zum Beispiel für Webserver mit *HTTPS*, zu erstellen, und zum anderen, um E-Mail-Zertifikate zu erstellen, die zum Beispiel von *S/MIME* eingesetzt werden können. Die zugrunde liegende Verzeichnisstruktur ist eine (praktische) Vorgabe von *OpenSSL*. Tabelle 31.4 zeigt die Intention, die dahintersteckt.

| Verzeichnis      | Typ         | Bedeutung                                              |
|------------------|-------------|--------------------------------------------------------|
| <i>certs</i>     | Verzeichnis | Speicherort der Zertifikate                            |
| <i>crl</i>       | Verzeichnis | Ablage der Zertifikatssperrlisten (CRL-Dateien)        |
| <i>newcerts</i>  | Verzeichnis | Ablage neu erstellter Zertifikate                      |
| <i>private</i>   | Verzeichnis | Speicherort der privaten Schlüssel                     |
| <i>index.txt</i> | Datei       | Indexdatei der erstellten Zertifikate dieser CA        |
| <i>serial</i>    | Datei       | ASCII-Textdatei, die die aktuelle Seriennummer enthält |

**Tabelle 31.4** CA-Unterverzeichnisse

Zusätzlich müssen in dem *CA*-Verzeichnis (*RootCA*, *ServerCA* und *EMailCA*) jeweils eine leere *index.txt* und eine mit einer *HEX*-Zahl gefüllte Datei *serial* vorhanden sein. Die Datei *index.txt* bietet eine Übersicht über die ausgestellten Zertifikate und enthält deren Seriennummer, Gültigkeitszeitraum und aktuellen Status.

Die Datei *serial* hingegen stellt einen Zählerstand dar, der pro ausgestelltem Zertifikat inkrementiert wird. Der Ausgangswert kann dabei jeder beliebige *HEX*-Wert sein – im Beispiel aus Listing 31.7 haben wir daher den Wert auf 00 gesetzt.

## OpenSSL-Konfiguration

Vor dem eigentlichen Erstellen der CA muss die OpenSSL-Konfigurationsdatei an Ihrer PKI angepasst werden. Die zentrale Konfigurationsdatei `/etc/ssl/openssl.cnf` ist in Sektionen unterteilt.

Passen Sie die Grundkonfiguration, die die einzige ohne eigene Sektion ist, an Ihrer Verzeichnisstruktur an:

```
HOME = /root/CA
```

### Listing 31.8 Grundkonfiguration

In der Standardkonfiguration ist lediglich eine CA-Sektion enthalten, die `CA_default`, die auch als Standard-CA in der Sektion `ca` über den Parameter `default_ca` gewählt ist. Beim Aufruf von OpenSSL ohne Angabe einer CA wird diese benutzt. Passen Sie diesen Parameter daher an die CA an, in der Sie die meisten Zertifikate erstellen werden.

Damit jede CA ihre eigene Konfiguration bekommt, werden die CAs in eigenen Sektionen angelegt. Erweitern Sie die `openssl.cnf` um je eine Sektion für die RootCA, die ServerCA und die EMailCA:

```
[ RootCA ]
dir           = /root/CA/RootCA           # Where everything is kept
certs        = $dir/certs                 # Where the issued certs are kept
crl_dir      = $dir/crl                   # Where the issued crl are kept
database     = $dir/index.txt            # database index file.
new_certs_dir = $dir/newcerts            # default place for new certs.
certificate  = $dir/RootCA.cacert.pem    # The CA certificate
serial       = $dir/serial                # The current serial number
crl          = $dir/crl.pem               # The current CRL
private_key  = $dir/private/RootCA.cakey.pem # The private key
RANDFILE    = $dir/private/.rand         # private random number file
x509_extensions = ca_cert                 # The extensions to add to the cert
name_opt     = ca_default                 # Subject Name options
cert_opt     = ca_default                 # Certificate field options
crl_extensions = crl_ext
default_days = 365                        # how long to certify for
default_crl_days= 30                      # how long before next CRL
default_md   = sha256                     # which md to use.
preserve     = no                         # keep passed DN ordering
policy       = policy_match
```

### Listing 31.9 Die Sektion »RootCA«

Tabelle 31.5 zeigt, welche Konfigurationen Sie pro CA anpassen müssen.

| Parameter       | Bedeutung                                                          |
|-----------------|--------------------------------------------------------------------|
| dir             | Root-Verzeichnis der CA                                            |
| certificate     | Root-Zertifikat der CA ( <i>RootCA.cacert.pem</i> )                |
| private_key     | privater Schlüssel der CA ( <i>private/RootCA.cakey.pem</i> )      |
| x509_extensions | Angabe der Sektion für die X.509-Erweiterungen der CA              |
| default_days    | Gültigkeitsdauer eines über die CA erstellten Zertifikats in Tagen |
| policy_match    | Definition der Regeln für diese CA                                 |

**Tabelle 31.5** Anzupassende Parameter pro CA

Einer der entscheidenden Parameter ist hierbei `x509_extensions`. Dieser verweist auf eine weitere Sektion, in der der Verwendungszweck der über diese CA erstellten Zertifikate spezifiziert ist. Erstellen Sie folgende Sektionen, denen Sie den jeweiligen Parameter `x509_extensions` zuweisen:

```
[ ca_cert ]
basicConstraints      = CA:TRUE
nsComment            = "OpenSSL Generated Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always

[ server_cert ]
basicConstraints      = CA:FALSE
nsComment            = "OpenSSL Generated Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage              = digitalSignature, keyEncipherment
extendedKeyUsage      = serverAuth

[ email_cert ]
basicConstraints      = CA:FALSE
nsComment            = "OpenSSL Generated Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage              = digitalSignature, keyEncipherment
extendedKeyUsage      = emailProtection
```

**Listing 31.10** Die Sektion »ca\_cert«

Beachten Sie, dass `basicConstraints` lediglich in der Sektion `ca_cert` auf `CA:TRUE` gesetzt ist. Dieser Parameter stellt sicher, dass lediglich über die RootCA CA-Zertifikate erstellt werden können. Ein weiterer wichtiger Parameter ist `policy_match`. Dieser definiert eine weitere Sektion, in der die Vorgaben für die Erstellung von Zertifikaten hinterlegt sind. Folgende Werte sind in der Standardkonfiguration enthalten:

```
countryName           = match
stateOrProvinceName  = match
organizationName     = match
organizationalUnitName = optional
commonName           = supplied
emailAddress         = optional
```

**Listing 31.11** Die Sektion »`policy_match`«

Die Wertigkeit der einzelnen Parameter kann wie folgt sein:

- ▶ `match`: muss identisch mit dem Wert des Stammzertifikats sein.
- ▶ `supplied`: muss angegeben sein, muss aber nicht mit dem Wert des Stammzertifikats übereinstimmen.
- ▶ `optional`: Dieser Parameter ist nicht weiter relevant – er kann auch ungesetzt sein.

Passen Sie abschließend in der Sektion `req_distinguished_name` die Standardwerte für die Zertifikatserstellung an Ihre Gegebenheiten an:

```
countryName_default      = DE
stateOrProvinceName_default = NRW
0.organizationName_default = Example Ltd
```

**Listing 31.12** Standardwerte bei der Zertifikatserstellung

Diese Werte werden Ihnen bei jeder Zertifikatserstellung als Default angeboten. Sie können sie durch die Eingabe von  auswählen.

### CA erzeugen

OpenSSL erwartet in jeder CA im Verzeichnis `private` eine Random-Datei mit dem Namen `.rand`. Diese Datei wird zur Unterstützung bei der Erstellung von Schlüsseln und Zertifikaten durch OpenSSL verwendet. Damit diese Werte definitiv auf jedem System einzigartig sind, müssen Sie sie vorab erzeugen. Erstellen Sie zunächst die Random-Dateien für jede CA:

```
root@example:~/CA# openssl rand -out RootCA/private/.rand 1024
root@example:~/CA# openssl rand -out ServerCA/private/.rand 1024
root@example:~/CA# openssl rand -out EMailCA/private/.rand 1024
```

**Listing 31.13** Random-Dateien erstellen



OpenSSL erstellt jeweils eine Datei `.rand` mit einer Größe von 1024 Bit. Zertifikate können über ein Passwort geschützt werden. Dies ist vor allem bei sensiblen Zertifikaten sinnvoll und bei solchen, die von Benutzern eingesetzt werden. Durch den Passwortschutz erhöhen Sie die Sicherheit um den konzeptionellen Schutz von »Besitz und Wissen«, der zum Beispiel bei rechtswirksamen Abwicklungen über ein Zertifikat vom Gesetzgeber gefordert ist. Das RootCA-Zertifikat stellt solch ein sensibles Zertifikat dar. Viele Empfehlungen zielen darauf ab, das RootCA-Zertifikat am besten auf einem System ohne Netzwerkanschluss und hinter einer abgeschlossenen Tür aufzubewahren. Diese auf den ersten Blick drastische Vorgabe ist aber durchaus gerechtfertigt.

Wird Ihr RootCA-Zertifikat korrumpiert, ist es jedermann möglich, beliebig viele Zertifikate in Ihrem Namen zu erstellen. Das Ausmaß an Möglichkeiten und Schaden ist fast undenkbar. Sichern Sie Ihr RootCA-Zertifikat daher unbedingt mit einem Passwort ab.



Die Erstellung von Zertifikaten erfolgt dabei stets in drei Schritten:

1. privaten Schlüssel erzeugen
2. Zertifikatsanfrage erstellen
3. Zertifikat zurückerhalten (signierter öffentlicher Schlüssel)

Erstellen Sie zunächst den privaten Schlüssel der RootCA:

```
root@example:~/CA/RootCA# openssl genrsa -aes256 -out private/RootCA.cakey.pem \
  -rand private/.rand 2048
1024 semi-random bytes loaded
Generating RSA private key, 2048 bit long modulus
.....+++++++
.....+++++++
e is 65537 (0x10001)
Enter pass phrase for private/RootCA.cakey.pem:
Verifying - Enter pass phrase for private/RootCA.cakey.pem:
```

**Listing 31.14** »RootCA«: privaten Schlüssel erzeugen

Mit dem Parameter `genrsa` erzeugen Sie ein *RSA Private Key*. Der Parameter `-aes256` weist dem privaten Schlüssel nicht nur das AES-Verschlüsselungsverfahren mit 256 Bit Schlüssellänge zu, sondern legt vor allem auch fest, dass dieser Schlüssel passwortgeschützt sein soll. Nach der Angabe des Speicherortes mit `-out` und der Angabe der Random-Datei mit `-rand` wird OpenSSL noch die Schlüssellänge von 2048 Bit mitgeteilt. Abschließend werden Sie aufgefordert, das Passwort für den privaten Schlüssel einzugeben und zu bestätigen. Erstellen Sie anschließend das RootCA-Zertifikat mit folgendem Befehl:

```
root@example:~/CA/RootCA# openssl req -new -x509 -days 1827 -set_serial 00 \
  -key private/RootCA.cakey.pem -out RootCA.cacert.pem
```

**Listing 31.15** »RootCA«: Zertifikat erzeugen

Der Befehl `openssl req` erstellt eine Zertifikatsanfrage. Da es sich beim RootCA-Zertifikat aber um das oberste Zertifikat dieser Vertrauenskette handelt, kann es von niemandem signiert und zurückgesandt werden. Daher erzeugt der Befehl keine Zertifikatsanfrage, sondern direkt das Zertifikat. Darüber hinaus muss sichergestellt werden, dass das RootCA-Zertifikat das erste ist, was Sie dadurch gewährleisten, dass Sie die hexadezimale Seriennummer mittels `-set_serial` auf 00 setzen.

### Unter-CAs erzeugen

Alle Unter-CAs werden nach dem gleichen Schema erstellt. Exemplarisch wird das Vorgehen anhand der *ServerCA* demonstriert. Wechseln Sie in das Verzeichnis `/root/CA/ServerCA`, und führen Sie folgenden Befehl aus:

```
root@example:~/CA/ServerCA# openssl genrsa -aes256 -out private/ServerCA.cakey.pem \
  -rand private/.rand 2048
1024 semi-random bytes loaded
Generating RSA private key, 2048 bit long modulus
.....+++
.....+++
e is 65537 (0x10001)
Enter pass phrase for private/ServerCA.cakey.pem:
Verifying - Enter pass phrase for private/ServerCA.cakey.pem:
```

#### Listing 31.16 »ServerCA«: privaten Schlüssel erstellen

Da es sich um eine CA handelt, wird diese ebenfalls wieder mit einem Passwort versehen (Parameter `-aes265`). Anschließend wird der *Signing Request* erzeugt:

```
root@example:~/CA/ServerCA# openssl req -new \
  -key private/ServerCA.cakey.pem -out ServerCA.cacert.req.pem
Enter pass phrase for private/ServerCA.cakey.pem:
```

#### Listing 31.17 »ServerCA«: Request erzeugen

Bestätigen Sie die Passwortabfrage mit dem vorher erstellten Passwort für den privaten Schlüssel der RootCA. Die erstellte Datei *ServerCA.cacert.req.pem* wird nun mit folgendem Befehl von der RootCA signiert:

```
root@example:~/CA/ServerCA# openssl ca -name RootCA \
  -in ServerCA.cacert.req.pem -out ServerCA.cacert.pem
Enter pass phrase for /root/CA/RootCA/private/RootCA.cakey.pem:
```

#### Listing 31.18 »ServerCA«: Zertifikat signieren

Wie bereits erläutert wurde, muss für die Signierung der »Nicht-Standard-CA« diese explizit angegeben werden. Dies wurde durch den Parameter `-name RootCA` erreicht.



Bedenken Sie, dass nun das Passwort des privaten Schlüssels der *RootCA* erfragt wird und nicht das Passwort der zuvor gesetzten *ServerCA*. Diese Passwörter sollten unbedingt unterschiedlich sein.

Beim Betrieb einer PKI sollten Sie einige Konventionen einhalten. Dazu zählt die Archivierung von Zertifikaten. Das soeben erstellte Zertifikat *ServerCA.cacert.pem* finden Sie ebenfalls, da es von der *RootCA* ausgestellt wurde, und zwar unter dem dafür vorgesehenen Verzeichnis */root/CA/RootCA/newcerts*. Die Namensgebung entspricht der Seriennummer des Zertifikats. Verschieben Sie dieses nach */root/CA/RootCA/certs/*, da es in Benutzung und nicht mehr »neu« ist. Im Betrieb sucht OpenSSL Zertifikate anhand ihres 8-Bit-Hash-Werts. Verlinken Sie daher das Zertifikat mit seinem Hash-Wert als Namen.

```
root@example:~/CA/RootCA/certs# ln -s 01.pem $(openssl x509 -hash -noout \
-in 01.pem).0
root@example:~/CA/RootCA/certs# ls -l
-rw-r--r-- 1 root root 4806 2018-04-28 19:25 01.pem
lrwxrwxrwx 1 root root    6 2018-04-28 20:37 b189e175.0 -> 01.pem
```

#### Listing 31.19 Zertifikat-Hash-Wert-Link erzeugen

Die Dateinamenerweiterung *.0* des Links stellt dabei eine Indizierung dar. Wird das Zertifikat neu herausgegeben, so wird dieser Wert inkrementiert. Wiederholen Sie diese Arbeitsschritte zur Erstellung der *EMailCA*.

### Zertifikate ausstellen

Das Erstellen von Zertifikaten erfolgt, wie bereits erläutert wurde, immer nach demselben Schema. Exemplarisch wurden in Listing 31.20 die Befehle zur Erstellung eines Serverzertifikats für *mail.example.com* aufgelistet:

```
root@example:~/CA/ServerCA# openssl genrsa -out mail.example.com.key.pem \
-rand ./private/.rand 1024

root@example:~/CA/ServerCA# openssl req -new -key mail.example.com.key.pem \
-out mail.example.com.req.pem

root@example:~/CA/ServerCA# openssl ca -name ServerCA \
-in mail.example.com.req.pem -out mail.example.com.cert.pem

root@example:~/CA/ServerCA# mv newcerts/00.pem certs/00.pem

root@example:~/CA/ServerCA# cd certs
root@example:~/CA/ServerCA/certs/# ln -s 00.pem $(openssl x509 -hash -noout \
-in 00.pem).0
```

#### Listing 31.20 Serverzertifikat erzeugen

Jedes über eine CA erstellte Zertifikat wird in der Datei *index.txt* notiert. Die Datei besteht aus sechs Feldern, die durch je einen Tabulator getrennt sind und dabei die in Tabelle 31.6 dargestellte Bedeutung haben.

| Feld-Nr. | Wert             | Bedeutung                                                                             |
|----------|------------------|---------------------------------------------------------------------------------------|
| 1        | V                | Zertifikatsstatus: V ( <i>valid</i> ), R ( <i>revoked</i> ) oder E ( <i>expired</i> ) |
| 2        | 240113154820Z    | UTZ-Zeit des Gültigkeitsendes des Zertifikats (Format: YYMMDDHHMMSSZ)                 |
| 3        | –                | Widerrufszeit (bei gültigen Zertifikaten leer)                                        |
| 4        | 01               | Seriennummer des Zertifikats                                                          |
| 5        | unknown          | Aufbewahrungsort des Zertifikats (wird derzeit immer mit unknown angegeben)           |
| 6        | /C=DE/ST=NRW/... | Subject-Line des Zertifikats                                                          |

**Tabelle 31.6** Felder der »index.txt«

Bei der Erstellung neuer Zertifikate wird jeweils die letzte Version der Dateien *index.txt* und *serial* mit der Endung *.old* versehen und bleibt erhalten.

### Zertifikate widerrufen

Beim Widerrufen von Zertifikaten, was man auch als *Sperren* bezeichnet, wird eine CRL (*Certificate Revocation List*) erstellt. Diese enthält Informationen zum Zertifikat und dazu, wann dieses für ungültig erklärt wurde. Dies ist sinnvoll, wenn ein Zertifikat verloren gegangen ist, gelöscht oder korrumpiert wurde. Neben den Namenskonventionen sieht der Betrieb einer CA auch das Pflegen der CRLs vor.

Sie müssen die CRLs erstellen und über einen Webserver zum Download zur Verfügung stellen. Um Zertifikate zu widerrufen, setzen Sie folgenden Befehl auf das betreffende Zertifikat ab:

```
root@example:~/CA/ServerCA# openssl ca -revoke certs/00.pem
Enter pass phrase for /root/CA/ServerCA/private/ServerCA.cakey.pem:
Revoking Certificate 00. Data Base Updated
```

#### Listing 31.21 Zertifikat widerrufen

OpenSSL setzt das Zertifikat in der Datenbank (*index.txt*) auf »R« (Revoke = Widerruf) und ergänzt das Feld »Widerrufszeit« (wie in Tabelle 31.6 beschrieben). Die benötigte Widerrufsliste erzeugen Sie mit nachstehendem Befehl:

```
root@example:~/CA/ServerCA# openssl ca -name ServerCA -gencrl -out crl/crl.pem
Enter pass phrase for /root/CA4/ServerCA/private/ServerCA.cakey.pem:
```

### Listing 31.22 Sperrlisten erzeugen

Die Informationen der CRL können Sie sich ebenfalls mit OpenSSL anzeigen lassen:

```
root@example:~/CA4/ServerCA# openssl crl -in crl/crl.pem -noout -text
Certificate Revocation List (CRL):
  Version 2 (0x1)
  Signature Algorithm: sha1WithRSAEncryption
  Issuer: /C=DE/ST=NRW/O=Example Ltd/CN=ServerCA/emailAddress=ca@example.com
  Last Update: Jan 13 15:53:01 2023 GMT
  Next Update: Feb 12 15:53:01 2023 GMT
  CRL extensions:
    X509v3 Authority Key Identifier:
      24:F7:6D:A9:96:53:49:FD:79:42:53:BB:F1:75:E3:BC:67:08:ED:D5
  Revoked Certificates:
    Serial Number: 00
      Revocation Date: Jan 13 15:52:29 2023 GMT
      Signature Algorithm: ...
```

### Listing 31.23 Sperrlisten anzeigen lassen

Unter dem Punkt Revoked Certificates werden alle widerrufenen Zertifikate mit ihrer Seriennummer und dem Datum ihres Widerrufs aufgelistet. Ein erneutes Erstellen der CRL erweitert die Einträge entsprechend.

### Weitere nützliche OpenSSL-Funktionen

Neben den bisher gezeigten OpenSSL-Befehlen gibt es eine Reihe weiterer und sehr nützlicher Befehle.

In Listing 31.24 finden Sie die nützlichsten zusammengefasst.

```
# Ausgabe des gesamten Inhalts
openssl x509 -text -noout -in <CERT>

# zeigt den Herausgeber an
openssl x509 -issuer -noout -in <CERT>

# Ausgabe des Verwendungszwecks
openssl x509 -purpose -noout -in <CERT>

# Ausgabe des "subject"
openssl x509 -subject -noout -in <CERT>
```

```
# Ausgabe des Gültigkeitszeitraums
openssl x509 -dates -noout -in <CERT>

# entfernt das Passwort eines privaten Schlüssels
openssl rsa -in <KEY> -out <KEY>

# fügt dem privaten Schlüssel ein Passwort hinzu
openssl rsa -aes256 -in <KEY> -out <KEY>
```

**Listing 31.24** OpenSSL-Befehle

### 31.5.2 Lokale Zertifikatsausstellung wie Let's Encrypt: *acme2certifier*

Das *Automatic Certificate Management Environment* (ACME) ist ein Protokoll zur automatischen Prüfung der Inhaberschaft einer Domain und dient der vereinfachten Ausstellung von Zertifikaten für die TLS-Verschlüsselung. Das Ziel dieser Umgebung ist es, die Zertifikate automatisiert und kostengünstig auszustellen. ACME wurde von der *Internet Security Research Group* (ISRG) für den Einsatz im Let's-Encrypt-Dienst definiert. Wenn Sie also schon mal ein Let's-Encrypt-Zertifikat angefordert haben, dann haben Sie bereits mit ACME gearbeitet.

Um über dieses Protokoll den Systemen in Ihrem Netzwerk Zertifikate Ihrer eigenen CA ausstellen zu lassen, können Sie das Projekt *acme2certifier* einsetzen. Mit dem in Python geschriebenen Tool können Sie unterschiedliche CAs einbinden und via ACME anbieten. Weitere Informationen finden Sie auf der Projektseite unter [github.com/grindsa/acme2certifier](https://github.com/grindsa/acme2certifier).

#### Voraussetzungen

Damit *acme2certifier* Zertifikate ausstellen kann, benötigt es eine CA. Wir verwenden dafür einfach die CA, die wir bereits in Abschnitt 31.5.1, »Einrichtung einer PKI mit Server- und E-Mail-Zertifikaten«, erstellt haben, und richten eine Unter-CA für die ACME-Zertifikate ein:

```
### Verzeichnisse und Dateien erstellen
ca:/root/CA# mkdir -p acmeCA/{certs,crl,newcerts,private}
ca:/root/CA# touch acmeCA/index.txt
ca:/root/CA# echo "00" > acmeCA/serial
ca:/root/CA# openssl rand -out acmeCA/private/.rand 1024

### Schlüssel, Zertifikat und Ausstellungsanfrage erstellen
ca:/root/CA# openssl genrsa -aes256 -out acmeCA/private/acmeCA.cakey.pem \
  -rand acmeCA/private/.rand 2048
ca:/root/CA # openssl req -new -x509 -days 1827 -set_serial 00 \
  -key acmeCA/private/acmeCA.cakey.pem -out acmeCA/acmeCA.cacert.pem
ca:/root/CA # openssl req -new -key private/acmeCA.cakey.pem \
  -out acmeCA.cacert.req.pem
```

```
### Mit der RootCA signieren
ca:/root/CA# openssl ca -name RootCA -in acmeCA/acmeCA.cacert.req.pem \
-out acmeCA/acmeCA.cacert.pem
```

**Listing 31.25** Unter-CA für ACME einrichten: »acmeCA«

Zusätzlich werden ein Webserver mit WSGI (*Web Server Gateway Interface*) und Python3 benötigt. Installieren Sie daher die benötigten Pakete so wie in Listing 31.26 dargestellt:

```
### Debian/Ubuntu:
$ sudo apt install apache2 libapache2-mod-wsgi-py3 python3-pip apache2-data git
```

```
### openSUSE:
leap:~ # zypper install apache2 apache2-mod_wsgi-python3 python3-pip git
```

```
### CentOS:
[root@centos ~]# dnf install httpd mod_wsgi python3 python3-pip git
```

**Listing 31.26** Installation der Voraussetzungen für »acme2certifier«

### Installation und Konfiguration

Da nun alle Vorbereitungen getroffen sind, können wir mit der eigentlichen Installation und Konfiguration des *acme2certifier* beginnen. Auf der Projektseite [github.com/grindsa/acme2certifier](https://github.com/grindsa/acme2certifier) finden Sie im Übrigen viele weitere Konfigurationsbeispiele für unterschiedliche CAs, Webserver und ACME-Clients. In Listing 31.27 werden die aktuelle Version des *acme2certifier* via git geladen, die fehlenden Python3-Abhängigkeiten mittels pip3 installiert und die Beispiel-Apache-Konfiguration übernommen:

```
ca:~# cd /opt
ca:/opt# git clone https://github.com/grindsa/acme2certifier
[...]

ca:/opt# cd acme2certifier/
ca:/opt/acme2certifier # pip3 install -r requirements.txt
[...]

ca:/opt/acme2certifier# cp examples/apache2/apache_wsgi.conf \
/etc/apache2/sites-available/acme2certifier.conf
### openSUSE: /etc/apache2/vhosts.d/acme2certifier.conf
### CentOS: /etc/httpd/conf.d/acme2certifier.conf
```

**Listing 31.27** Installation von »acme2certifier«

Die Apache-Konfiguration ist bereits vollständig, Sie müssen eventuell nur die Pfade entsprechend Ihrer Distribution anpassen (openSUSE: `/srv/www`).

```
# a2enmod cgi
# a2enmod rewrite
<VirtualHost *:80>
    DocumentRoot /var/www/acme2certifier/
    WSGIDaemonProcess acme_srv python-path=/var/www/acme2certifier
    WSGIProcessGroup acme_srv
    WSGIApplicationGroup %GLOBAL
    WSGIScriptAlias / /var/www/acme2certifier/acme2certifier_wsgi.py
    <Directory /var/www/acme2certifier>
        Order allow,deny
        Allow from all
    </Directory>
</VirtualHost>
```

**Listing 31.28** Inhalt von »acme2certifier.conf«

Wie Sie in Listing 31.28 sehen, setzt *acme2certifier* zwei Apache-Module voraus – was dezent durch die abzusetzenden Kommandos in den ersten beiden Zeilen der Konfigurationsdatei dargestellt wird. Darum werden wir uns im weiteren Verlauf kümmern.

Nun legen wir aber zunächst die benötigten Verzeichnisse und Dateien an und legen dort die benötigten Inhalte ab (siehe Listing 31.29):

```
### Serverprogramm kopieren
ca:/opt/acme2certifier# mkdir /var/www/acme2certifier
ca:/opt/acme2certifier# cp examples/acme2certifier_wsgi.py \
/var/www/acme2certifier

### Beispiele kopieren
ca:/opt/acme2certifier# mkdir /var/www/acme2certifier/examples
ca:/opt/acme2certifier# cp -R examples/ca_handler/ \
/var/www/acme2certifier/examples/ca_handler

ca:/opt/acme2certifier# cp -R examples/db_handler/ \
/var/www/acme2certifier/examples/db_handler

ca:/opt/acme2certifier# cp -R examples/eab_handler/ \
/var/www/acme2certifier/examples/eab_handler

ca:/opt/acme2certifier# cp -R examples/hooks/ \
/var/www/acme2certifier/examples/hooks

ca:/opt/acme2certifier# cp -R examples/acme_srv.cfg \
/var/www/acme2certifier/examples/
```



```

### Hilfswerkzeuge kopieren
ca:/opt/acme2certifier# cp -R tools/ /var/www/acme2certifier/tools

### Funktionen kopieren
ca:/opt/acme2certifier# mkdir /var/www/acme2certifier/acme_srv
ca:/opt/acme2certifier# cp -R acme_srv/ /var/www/acme2certifier/

```

**Listing 31.29** Dateien und Verzeichnisse kopieren

Mit den Kommandos aus Listing 31.29 werden das eigentliche Serverprogramm (WSGI), die benötigten Beispiele aus dem Verzeichnis *examples*, die benötigten Hilfswerkzeuge und die eigentlichen ACME-Funktionen an ihren jeweiligen Bestimmungsort gebracht.

Anschließend können Sie nun die CA einrichten. Um die bereits eingerichtete OpenSSL-CA weiterverwenden zu können, müssen Sie die Kommandos aus Listing 31.30 absetzen:

```

### OpenSSL- und DB-Handler
ca:/var/www/acme2certifier# cp examples/ca_handler/openssl_ca_handler.py \
  acme_srv/ca_handler.py

ca:/var/www/acme2certifier# cp examples/db_handler/wsgi_handler.py \
  acme_srv/db_handler.py

### Konfigurationsdatei
ca:/var/www/acme2certifier# cp examples/acme_srv.cfg acme_srv/

### Zertifikate
ca:/var/www/acme2certifier# mkdir -p acme_srv/ca/certs
ca:/var/www/acme2certifier# cp /root/CA/acmeCA/private/acmeCA.cakey.pem acme_srv/ca/
ca:/var/www/acme2certifier# cp /root/CA/acmeCA/acmeCA.cacert.pem acme_srv/ca/
ca:/var/www/acme2certifier# cp /root/CA/RootCA/RootCA.cacert.pem acme_srv/ca/
ca:/var/www/acme2certifier# touch acme_srv/ca/crl.pem

```

**Listing 31.30** Die CA einrichten

Nun müssen Sie die soeben angelegte Konfigurationsdatei *acme\_srv.cfg* noch anpassen. In Listing 31.31 sind die notwendigen Anpassungen in Fettschrift dargestellt:

```

[CAhandler]
issuing_ca_key: /var/www/acme2certifier/acme_srv/ca/acmeCA.cakey.pem
issuing_ca_key_passphrase: <PASSHRASE>
issuing_ca_cert: /var/www/acme2certifier/acme_srv/ca/acmeCA.cacert.pem
issuing_ca_crl: /var/www/acme2certifier/acme_srv/ca/crl.pem
cert_validity_days: 30
cert_save_path: /var/www/acme2certifier/acme_srv/ca/certs
ca_cert_chain_list: ["RootCA.cacert.pem"]

```

```
openssl_conf: /var/www/acme2certifier/acme_srv/ca/openssl.conf
allowed_domainlist: ["foo.bar\\$", "foo1.bar.local"]
blocked_domainlist: ["google.com.foo.bar\\$", "host.foo.bar$", "\\*.foo.bar"]
save_cert_as_hex: True
cn_enforce: True
```

**Listing 31.31** Anpassungen an der Konfigurationsdatei »acme\_srv.cfg«

Wie Sie in Listing 31.31 sehen, wurden das Zertifikat, der Schlüssel und die dazugehörige Passphrase der CA angegeben. Zusätzlich wurde noch das Zertifikat der Root-CA mitangegeben, damit den ausgestellten Zertifikaten auch die Zertifikatskette beigelegt werden kann.

Die folgenden Direktiven sind in der Konfigurationsdatei zulässig und haben dabei folgende Funktionen:

- ▶ `issuing_ca_key` – Schlüssel der ausstellenden CA im PEM-Format
- ▶ `issuing_ca_key_passphrase` – Passphrase des CA-Schlüssels; falls Sie keinen vergeben haben, müssen Sie den Wert leer lassen.
- ▶ `[optional] issuing_ca_key_passphrase_variable` – Gibt den Namen einer Umgebungsvariablen an, die die Passphrase für den CA-Schlüssel enthält – eine mit `issuing_ca_key_passphrase` angegebene Passphrase wird bevorzugt.
- ▶ `issuing_ca_cert` – Zertifikat der ausstellenden CA im PEM-Format
- ▶ `issuing_ca_crl` – CRL der ausstellenden CA im PEM-Format
- ▶ `ca_cert_chain_list` – Liste der Root- und Unter-Zertifikate, die zur Zertifikatskette gehören (Die ausstellende CA muss hier nicht erneut angegeben werden.)
- ▶ `[optional] cert_validity_days` – Gültigkeitsdauer der ausgestellten Zertifikate in Tagen (standardmäßig 365); diese sollten Sie unbedingt verkürzen.
- ▶ `[optional] cert_save_path` – das Verzeichnis, in dem die ausgestellten Zertifikate gespeichert werden.
- ▶ `[optional] openssl_conf` – OpenSSL-Konfigurationsdatei mit der Sektion *extensions*. Sie enthält alle Extensions, die bei der Zertifikatsausstellung angewandt werden.
- ▶ `[optional] allowed_domainlist` – Liste der Domänen und Namen, für die Zertifikate ausgestellt werden dürfen. Die Angabe erfolgt durch reguläre Ausdrücke im JSON-Format.
- ▶ `[optional] blocked_domainlist` – Liste der Domänen und Namen, für die keine Zertifikate ausgestellt werden dürfen. Die Angabe erfolgt durch reguläre Ausdrücke im JSON-Format.
- ▶ `[optional] save_cert_as_hex` – Seriennummer im Hexadezimal-Format, die als Dateinamen zur Speicherung der ausgestellten Zertifikate verwendet werden (standardmäßig deaktiviert).
- ▶ `[optional] cn_enforce` – Setze den ersten Eintrag des *SubjectAlternativeNames* (SAN) als CN, falls keiner gesetzt ist (standardmäßig deaktiviert).

Sie sollten mindestens noch einen Blick auf die Direktive `allowed_domainlist` werfen. Da Sie das *RootCA*-Zertifikat Ihrer CA auf allen Clients in Ihrem Netzwerk verteilt haben, würden diese Clients allen von der *acmeCA* ausgestellten Zertifikaten vertrauen. Wirklich allen! Daher sollten Sie die Domänen, für die Zertifikate via ACME angefordert werden dürfen, immer auf Ihre interne Domäne beschränken.



Die in der Konfiguration angegebene Datei für die OpenSSL-Erweiterungen müssen wir jetzt noch mit dem Inhalt aus Listing 31.32 anlegen:

```
[extensions]
subjectKeyIdentifier    = hash, issuer:always
keyUsage                = digitalSignature, keyEncipherment
basicConstraints        = critical, CA:FALSEerr
authorityKeyIdentifier  = keyid:always, issuer:always
extendedKeyUsage        = critical, clientAuth, serverAuth
```

**Listing 31.32** Inhalt von »`/var/www/acme2certifier/acme_srv/ca/openssl.conf`«

Bitte beachten Sie, dass Sie hier nicht die eigentliche *openssl.cnf* Ihrer OpenSSL-CA angeben dürfen – dies würde zu Fehlern beim Start des *acme2certifier* führen.



Anschließend müssen Sie noch die Zugriffsrechte und die Eigentümer korrigieren, damit *acme2certifier* seinen Dienst aufnehmen kann (siehe Listing 31.33):

```
### Debian/Ubuntu
chown -R www-data.www-data /var/www/acme2certifier/
chmod a+x /var/www/acme2certifier/acme_srv

### openSUSE
chown -R wwwrun.wwwrun /srv/www/acme2certifier/
chmod a+x /srv/www/acme2certifier/acme_srv

### CentOS
chown -R apache.apache /var/www/acme2certifier/
chmod a+x /var/www/acme2certifier/acme_srv
```

**Listing 31.33** Anpassung der Dateirechte

Zu guter Letzt aktivieren Sie die bereits erwähnten und benötigten Apache-Module (siehe Listing 31.34):

```
### Debian/Ubuntu
a2enmod cgi rewrite

### openSUSE
a2enmod cgi rewrite mod_access_compat
```

```
### CentOS  
# -
```

**Listing 31.34** Webserver-Module aktivieren

Starten Sie nun den Webserver einmal neu, damit alle Anpassungen aktiv werden. Um zu überprüfen, ob alles funktioniert, genügt es, mit dem `curl`-Kommando den `localhost` nach dem Verzeichnis `/directory` zu fragen, so wie in Listing 31.35 dargestellt. Die Parameter `-s` und die Pipe vor `jq` dienen im Übrigen lediglich der besseren Lesbarkeit der Ausgabe.

```
$ curl -s http://127.0.0.1/directory | jq  
{  
  "newAuthz": "http://127.0.0.1/acme/new-authz",  
  "newNonce": "http://127.0.0.1/acme/newnonce",  
  "newAccount": "http://127.0.0.1/acme/newaccount",  
  "newOrder": "http://127.0.0.1/acme/neworders",  
  "revokeCert": "http://127.0.0.1/acme/revokecert",  
  "keyChange": "http://127.0.0.1/acme/key-change",  
  "meta": {  
    "home": "https://github.com/grindsa/acme2certifier",  
    "author": "grindsa <grindelsack@gmail.com>",  
    "name": "acme2certifier",  
    "version": "0.24"  
  },  
  "c701[...]69f62": "https://community.letsencrypt.org/t/[...]/33417"  
}
```

**Listing 31.35** Überprüfen, ob alles funktioniert, mit »curl«

Sieht die Ausgabe so aus wie in Listing 31.35, läuft Ihr `acme2certifier` einwandfrei. Und wie rufen Sie jetzt automatisiert Zertifikate ab? Dafür können Sie einfach den von Ihnen favorisierten ACME-Client verwenden. In Listing 31.36 sehen Sie zum Beispiel die benötigten Kommandos für den bereits vorgestellten `certbot` und zusätzlich die Befehle für das `bash`-Skript `acme.sh` (siehe [github.com/acmesh-official/acme.sh](https://github.com/acmesh-official/acme.sh)):

```
### certbot  
# Account anlegen  
$ certbot register --agree-tos -m daniel@example.com \  
  --server http://acme.example.com --no-eff-email  
  
# Zertifikat für "test.example.com" anfordern  
$ certbot certonly --server http://acme.example.com --standalone \  
  --preferred-challenges http -d test.example.com --cert-name test.example.com  
  
### acme.sh
```

```
# Account anlegen
acme.sh --server http://acme.example.com --register-account \
--accountemail daniel@example.com

# Zertifikat für "test.example.com" anfordern
acme.sh --server http://acme.example.com --issue -d test.example.com --standalone
```

**Listing 31.36** Anfordern eines Zertifikats via »certbot« und »acme.sh«

Wie Sie in Listing 31.36 sehen, müssen Sie, genau wie bei Let's Encrypt auch, zunächst einen Account anlegen und den Nutzungsbedingungen zustimmen. Anschließend können Sie Zertifikate wie gewohnt anfordern. Im Übrigen unterstützt *acme2certifier* auch die anderen Authentifizierungsmethoden von ACME, z. B. DNS-01.

### Debug-Tipps

Falls Ihre Konfiguration nicht auf Anhieb läuft, sollten Sie zunächst einen Blick in das *error.log* des Apache werfen. Dort protokolliert *acme2certifier* alle Zugriffe und auch Fehler, die auftreten. Wenn Sie dennoch die Ursache nicht finden können, dann können Sie mit dem Schalter `debug: true` in der Sektion [DEFAULT] der *acme\_srv.cfg* den Umfang der Protokollierung erweitern.

31

### 31.5.3 E-Mail-Verschlüsselung

Zur Absicherung des E-Mail-Verkehrs finden zwei gängige Methoden Anwendung:

- ▶ **GnuPG (GNU Privacy Guard)**  
GnuPG dient zum Ver- und Entschlüsseln von Daten sowie zum Erzeugen und Prüfen elektronischer Signaturen. GnuPG wurde als Ersatz für das unter Patent stehende PGP (*Pretty Good Privacy*) entwickelt und verwendet das in Abschnitt 31.3.5, »Ein anderer Ansatz: PGP (Web-of-Trust)«, besprochene Prinzip des *Web-of-Trust*.
- ▶ **S/MIME (Secure/Multipurpose Internet Mail Extensions)**  
S/MIME ist ein Standard zum Verschlüsseln und Signieren von MIME-gekapselten E-Mails und basiert auf dem PKI-Prinzip.

#### GnuPG

GnuPG wird vermehrt von Privatanwendern genutzt – allein schon aufgrund des Konzepts des *Web-of-Trust*, in dem jeder jeden über Ecken kennt. Nichtsdestotrotz stellt es eine gut implementierte Lösung zum Signieren und Verschlüsseln von E-Mails dar. Auch dieses Verfahren setzt auf Zertifikate mit einem öffentlichen und einem privaten Schlüssel. Lediglich die übergeordnete Kopfstelle existiert nicht, da die Authentizität zwischen den Protagonisten selbst geregelt wird. Durch das Grundprinzip der Vertrauenskette gibt es unterschiedliche Vertrauensstufen. In Tabelle 31.7 sehen Sie die Vertrauensstufen, die Sie vergeben können.

| Stufe                   | Bedeutung                                          |
|-------------------------|----------------------------------------------------|
| unbekannt               | keine Angaben vorhanden                            |
| kein Vertrauen          | Der Signatur wird NIE vertraut.                    |
| geringfügiges Vertrauen | Personen, die man nicht persönlich kennt           |
| volles Vertrauen        | Personen, denen man vollkommen vertraut            |
| ultimatives Vertrauen   | Diese Stufe findet nur bei einem selbst Anwendung. |

**Tabelle 31.7** GnuPG-Vertrauensstatus

Bei GnuPG findet der Begriff *Schlüsselbund* Anwendung. Dieser bezieht sich auf die Speicherform der Schlüssel in einem »Klumpen«, also in einer Datei, und nicht – wie es bei einer PKI üblich ist – je Schlüssel/Zertifikat. Bevor Sie anderen das Vertrauen aussprechen können, benötigen Sie zunächst selbst einen Schlüssel. Diesen können Sie mit GnuPG wie folgt erzeugen:

```
daniel@example:~# gpg --full-generate-key
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

```
gpg: Verzeichnis '/home/daniel/.gnupg' erzeugt
gpg: Die "keybox" '/home/daniel/.gnupg/pubring.kbx' wurde erstellt
```

**Listing 31.37** GnuPG-Schlüssel erzeugen

Bei der ersten Verwendung von *gpg* werden die Verzeichnisse und benötigten Dateien vorab erstellt. Wählen Sie anschließend den Verwendungszweck und den Verschlüsselungsstandard (siehe Listing 31.38). Der Wert *RSA* und *RSA* entspricht dem aktuellen Standard.

Bitte wählen Sie, welche Art von Schlüssel Sie möchten:

- (1) RSA und RSA (voreingestellt)
- (2) DSA und Elgamal
- (3) DSA (nur signieren/beglaubigen)
- (4) RSA (nur signieren/beglaubigen)
- (14) Vorhandener Schlüssel auf der Karte

Ihre Auswahl? **1**

**Listing 31.38** GnuPG: Auswahl der Schlüsselart

Wählen Sie anschließend die Länge des Schlüssels (siehe Listing 31.39). Der Standardwert von 2048 Bit Länge entspricht dem aktuellen Standard bei Verschlüsselungsverfahren.

RSA-Schlüssel können zwischen 1024 und 4096 Bit lang sein.  
Welche Schlüssellänge wünschen Sie? (3072)

**Listing 31.39** GnuPG: Auswahl der Schlüssellänge

Wählen Sie nun die Gültigkeitsdauer des Schlüssels (siehe Listing 31.40). Bedenken Sie, dass ein verloren gegangener Schlüssel zwar widerrufen werden kann, dass mit ihm aber weiterhin gültig verschlüsselt werden kann.

Widerstehen Sie daher der Versuchung, den Schlüssel mit unbegrenzter Gültigkeit zu erstellen, und wählen Sie besser einen Wert zwischen zwei und vier Jahren.



Bitte wählen Sie, wie lange der Schlüssel gültig bleiben soll.

```
0 = Schlüssel verfällt nie
<n> = Schlüssel verfällt nach n Tagen
<n>w = Schlüssel verfällt nach n Wochen
<n>m = Schlüssel verfällt nach n Monaten
<n>y = Schlüssel verfällt nach n Jahren
```

Wie lange bleibt der Schlüssel gültig? (0)

```
Schlüssel verfällt nie
Ist dies richtig? (j/N) j
```

**Listing 31.40** GnuPG: Auswahl der Gültigkeitsdauer

Auch zu einem *gpg*-Schlüssel gehören Benutzerangaben. Geben Sie diese entsprechend der Aufforderung an:

GnuPG erstellt eine User-ID, um Ihren Schlüssel identifizierbar zu machen.

```
Ihr Name ("Vorname Nachname"): Max Mustermann
Email-Adresse: mm@example.com
Kommentar:
Sie haben diese User-ID gewählt:
  "Max Mustermann <mm@example.com>"
```

Ändern: (N)ame, (K)ommentar, (E)-Mail oder (F)ertig/(A)bbrechen? F

**Listing 31.41** GnuPG: Angaben zur User-ID

Der private Schlüssel muss mit einem Passwort geschützt werden. Vergeben Sie dieses, und bestätigen Sie es:

Bitte geben Sie die Passphrase ein,  
um Ihren Schlüssel zu schützen.

Passphrase: \_\_\_\_\_

<OK>

<Abbrechen>

#### Listing 31.42 GnuPG: Passwort festlegen

Anders als bei der Schlüsselerzeugung mithilfe von OpenSSL wird hier keine *Random-Datei* verwendet, sondern es werden Zufallswerte aus der gerade laufenden Sitzung extrahiert.

Dazu müssen Sie GnuPG unterstützen, wozu Sie auch aufgefordert werden. Geben Sie in einem weiteren Fenster »wilde« Zeichenfolgen ein, und/oder bewegen Sie die Maus.

Wir müssen eine ganze Menge Zufallswerte erzeugen. Sie können dies unterstützen, indem Sie z.B. in einem anderen Fenster/Konsole irgendetwas tippen, die Maus verwenden oder irgendwelche anderen Programme benutzen.

```
gpg: /home/daniel/.gnupg/trustdb.gpg: trust-db erzeugt
gpg: Schlüssel 91A5382D5C45AFFC ist als ultimativ vertrauenswürdig gekennzeichnet
gpg: Verzeichnis `/home/daniel/.gnupg/openpgp-revocs.d' erzeugt
gpg: Widerrufszertifikat wurde als '/home/daniel/.gnupg/openpgp-revocs.d/81C1954C\
7F338CC61C7A549891A5382D5C45AFFC.rev' gespeichert.
Öffentlichen und geheimen Schlüssel erzeugt und signiert.
```

```
pub  rsa3072 2023-01-13 [SC]
     81C1954C7F338CC61C7A549891A5382D5C45AFFC
uid                               Max Mustermann <mm@example.com>
sub  rsa3072 2023-01-13 [E]
```

#### Listing 31.43 GnuPG: Erzeugung von Zufallswerten

##### **Fehler: »gpg: agent\_genkey failed: Kein Pinentry«**

Falls Sie bei der Erstellung eine Fehlermeldung erhalten, sollten Sie prüfen, ob das Paket *pinentry* installiert ist.

Falls die Erstellung dennoch fehlschlägt, versuchen Sie es erneut ohne SSH – melden Sie sich also »echt« an der Konsole des Systems an; und falls auch das fehlschlägt, ergänzen Sie beim Kommando den Parameter `--pinentry-mode loopback`. Damit erfolgt die Abfrage des Passworts direkt auf der Konsole.

Sind ausreichend viele Zufallszahlen generiert worden, erstellt GnuPG Ihren privaten Schlüssel und legt weitere noch nicht vorhandene Dateien an. Alle von GnuPG angelegten Daten befinden sich im Homeverzeichnis unter dem Verzeichnis *.gnupg*. Die in Tabelle 31.8 dargestellten Dateien sind dort hinterlegt.



| Datei                     | Bedeutung                                               |
|---------------------------|---------------------------------------------------------|
| <i>pubring.kbx</i>        | Schlüsselbund der öffentlichen Schlüssel und Signaturen |
| <i>trustdb.gpg</i>        | Vertrauensstufe für fremde öffentliche Schlüssel        |
| <i>private-keys-v1.d/</i> | Verzeichnis mit dem/den eigenen privaten Schlüssel(n)   |
| <i>openpgp-revocs.d/</i>  | Verzeichnis mit den Widerrufsschlüsseln                 |

**Tabelle 31.8** GnuPG-Dateien

Entscheidend für das Prinzip des Web-of-Trust ist das Publizieren und Verteilen des eigenen Schlüssels. Exportieren Sie daher als Erstes Ihren öffentlichen Schlüssel:

```
daniel@example:~# gpg --output max_mustermann.gpg --export mm@example.com
```

**Listing 31.44** GnuPG: Schlüssel exportieren

31

GnuPG erzeugt nun eine binäre Schlüsseldatei. Der Schlüssel kann ebenfalls als Klartext ausgegeben werden. Verwenden Sie dafür den Schalter `-a`. Sie können Ihren öffentlichen Schlüssel nun auf Ihrer Website zum Download anbieten oder ihn per E-Mail an Ihre Bekannten verschicken. Erhalten Sie Schlüssel von Kommunikationspartnern, müssen Sie diese in Ihren Public-Schlüsselbund importieren (siehe Listing 31.45):

```
daniel@example:~# gpg --import Ute_Musterfrau.gpg
```

**Listing 31.45** GnuPG: Schlüssel importieren

Dieser Schlüssel wird somit Ihrem »Schlüsselbund« hinzugefügt, sodass dieser durch GnuPG ausgelesen werden kann, um Nachrichten zu verifizieren. Zusätzlich kann diesem Schlüssel jetzt noch das Vertrauen ausgesprochen werden. Dafür bietet GnuPG einen separaten Befehlsmodus. Starten Sie GnuPG mit dem folgenden Parameter:

```
daniel@example:~# gpg --edit-key um@example.com
[...]
gpg: "Trust-DB" wird überprüft
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: Tiefe: 0 gültig: 1 signiert: 0 Vertrauen: 0-, 0q, 0n, 0m, 0f, 1u
pub rsa3072/DD0E1DC4D6FCF40D
    erzeugt: 2023-01-13 verfällt: niemals    Nutzung: SC
    Vertrauen: unbekannt    Gültigkeit: unbekannt
sub rsa3072/8B77D4FD6E0B1826
    erzeugt: 2023-01-13 verfällt: niemals    Nutzung: E
[ unbekannt ] (1). Ute Musterfrau <um@example.com>
```

**Listing 31.46** GnuPG: einem Schlüssel das Vertrauen aussprechen

Der Schlüssel der Person »Ute Musterfrau« steht sowohl bei Vertrauen als auch bei Gültigkeit auf unbekannt. Das Vertrauen können Sie selbst bestimmen. Verwenden Sie dafür den Befehl `trust`. Anschließend können Sie die Vertrauensstufe zu diesem Schlüssel einstellen. Neben der Möglichkeit, Schlüssel manuell auszutauschen, über E-Mail, einen USB-Stick oder – wie es eine Zeit lang in »Mode« war – auf einer sogenannten *Keysigning-Party*, können Sie diese auch über *Keyserver* publizieren.

### E-Mail-Verschlüsselung testen

Jeder moderne *MTA*<sup>10</sup> bietet die Möglichkeit, GnuPG-Schlüssel zu verarbeiten. Dazu gehören zum Beispiel:

- ▶ *Evolution*
- ▶ *KMail*
- ▶ *Mozilla Thunderbird*

Diese MTAs ermöglichen es Ihnen, komfortabel Ihren Schlüssel und die Schlüssel Ihrer Kommunikationspartner hinzuzufügen und zu verwalten. Damit Sie den erstellten Schlüssel testen können, hat die *Free Software Foundation* im Projekt *EMAIL SELF-DEFENSE* (siehe *email-selfdefense.fsf.org*) den E-Mail-Roboter »Edward« geschaffen.

Senden Sie eine unverschlüsselte E-Mail mit Ihrem öffentlichen Schlüssel (entweder als Klartext in der Mail oder als Anhang) mit dem Betreff »Mein öffentlicher Schlüssel« an *edward-en@fsf.org*. Der E-Mail-Roboter sendet Ihnen daraufhin eine mit Ihrem öffentlichen Schlüssel verschlüsselte E-Mail zurück. In dieser ist der öffentliche Schlüssel von Edward enthalten.

Importieren Sie diesen Schlüssel. Dann können Sie Edward selbst eine mit Ihrem öffentlichen Schlüssel verschlüsselte Nachricht senden, die er ebenfalls wieder beantwortet. Erhalten Sie keine Antwort, sollten Sie Ihren Schlüssel noch einmal genauer unter die Lupe nehmen.

### Keyserver

Wie bereits erläutert wurde, findet die effektivste Art des Schlüsselaustauschs über *Keyserver* statt. Es existiert eine Vielzahl solcher Server; nachstehend eine Auswahl:

- ▶ *keyserver.pgp.com*
- ▶ *keys.openpgp.org*
- ▶ *pgp.uni-mainz.de*
- ▶ *pgp.mit.edu*

---

<sup>10</sup> *Mail Transfer Agent*, Software zum Versand von E-Mails



### Hochladen von Schlüsseln

Es sei nochmals eindringlich darauf hingewiesen, dass Sie Ihren Schlüssel mit Bedacht publizieren sollten, da Sie einmal hochgeladene Schlüssel nicht einfach entfernen, sondern nur widerrufen können.

Welchen Server Sie verwenden, spielt prinzipiell keine Rolle, da die Keyserver untereinander einen regen Austausch betreiben. Tabelle 31.9 zeigt, welche Befehle beim Umgang mit Keyservern relevant sind.

| Befehl                                             | Bedeutung                   |
|----------------------------------------------------|-----------------------------|
| <code>gpg ---send-key &lt;KEY-ID&gt;</code>        | Schlüssel hochladen         |
| <code>gpg ---search-keys "Vorname Nachname"</code> | nach einem Schlüssel suchen |
| <code>gpg ---recv-keys &lt;KEY-ID&gt;</code>       | Schlüssel herunterladen     |
| <code>gpg ---refresh-keys &lt;KEY-ID&gt;</code>    | Schlüssel aktualisieren     |

**Tabelle 31.9** »gpg«-Befehle

### Widerrufszertifikate

Ist das Kind dann doch in den Brunnen gefallen, hilft nur noch eines: der Widerruf. Da GnuPG auf das Web-of-Trust setzt, sollten Sie den Widerruf nicht nur zum Keyserver senden, sondern auch alle Ihre Kontakte benachrichtigen. Ein Widerrufszertifikat erstellen Sie mit der Option `---gen-revoke` und der Key-ID des betreffenden Schlüssels, die Sie als Parameter übergeben müssen. Entspricht die Ausgabe von GnuPG dem Schlüssel, den Sie widerrufen möchten, fahren Sie mit der Eingabe von `y` fort (siehe Listing 31.47):

```
daniel@example:~# gpg --output revoke_F1FFA254.asc --gen-revoke [...]F1FFA254
```

```
sec  rsa3072/BEC7D05BF1FFA254 2023-01-13 Max Mustermann <mm@example.com>
```

Ein Widerrufszertifikat für diesen Schlüssel erzeugen? (j/N)

**Listing 31.47** GnuPG: Widerrufszertifikat erzeugen

Anschließend werden Sie aufgefordert, den Grund für den Widerruf anzugeben. Wählen Sie den entsprechenden Grund aus, und geben Sie, falls notwendig, noch eine Beschreibung an:

Grund für den Widerruf:

0 = Kein Grund angegeben

1 = Hinweis: Dieser Schlüssel ist nicht mehr sicher

```
2 = Schlüssel ist überholt
3 = Schlüssel wird nicht mehr benutzt
Q = Abbruch
(Wahrscheinlich möchten Sie hier 1 auswählen)
Ihre Auswahl? 1
Geben Sie eine optionale Beschreibung ein. Beenden mit einer leeren Zeile:
> Laptop Diebstahl
>
Grund für Widerruf: Hinweis: Dieser Schlüssel ist nicht mehr sicher
Laptop Diebstahl
Ist das OK? (j/N) j
```

**Listing 31.48** GnuPG: Grund des Widerrufs

Anschließend müssen Sie das Passwort des Schlüssels angeben:

```
Sie benötigen eine Passphrase, um den geheimen OpenPGP Schlüssel zu entsperren:
"Max Mustermann <mm@example.com>"
3072-Bit RSA Schlüssel, ID BEC7D05BF1FFA254,
erzeugt 2023-01-13.
```

**Listing 31.49** GnuPG: Passwortabfrage**Der Haken bei der Erstellung von Widerrufszertifikaten**

Zum Erstellen von Widerrufszertifikaten müssen Sie auch über das Zertifikat selbst verfügen. Sie können nur dann ein Widerrufszertifikat erstellen, wenn Sie den entsprechenden Schlüssel auch besitzen! Legen Sie das Widerrufszertifikat also am besten direkt nach der Erstellung an, und sichern Sie dieses gewissenhaft.

Das erstellte Widerrufszertifikat kann nun publiziert werden. Fügen Sie dieses auch Ihrem Schlüsselbund hinzu, damit Sie nicht Opfer Ihres eigenen Schlüssels werden:

```
daniel@example:~# gpg --import Widerruf_BBF427A7.asc
```

**Listing 31.50** GnuPG: Widerrufszertifikat importieren**S/MIME**

Über *S/MIME* können E-Mails ähnlich wie bei GnuPG signiert und verschlüsselt werden. Hierfür werden zwei Content-Typen zur Verfügung gestellt: zum einen das *Multipart/Signed-Format* zur Signierung und zum anderen das *Multipart/Encrypted-Format* zur Verschlüsselung. Anders als bei GnuPG basiert S/MIME auf dem PKI-Prinzip. Daher findet dieses Modell auch eher in Unternehmen Anwendung. Über (kostenpflichtige) öffentliche

Zertifikate von TrustCentern können E-Mails ohne Zutun des Nutzers authentifiziert und entschlüsselt werden. Sobald Sie über ein X.509-Zertifikat verfügen, können Sie über alle gängigen MTAs die S/MIME-Signierung/Verschlüsselung konfigurieren.

## 31.6 Neben der Kommunikation – Dateiverschlüsselung

Das zweite große Anwendungsgebiet der Verschlüsselung neben der Kommunikation ist die Dateiverschlüsselung. Diese erfolgt lokal, und außer einzelnen Dateien können Sie auch Partitionen oder das ganze System verschlüsseln.

### 31.6.1 Dateien

Eine einfache Methode der Dateiverschlüsselung bietet uns GnuPG. Mit dem Toolkit können Sie Dateien zum Beispiel auch mit einem *Pre-Shared Key* (PSK) verschlüsseln:

```
daniel@example:~# gpg -c datei.txt
```

**Listing 31.51** Mit einem PSK verschlüsseln

Durch das Argument `--cipher-algo` kann der Verschlüsselungsalgorithmus bestimmt werden. Standardmäßig wird hier *AES256* verwendet. Eine Liste der unterstützten Algorithmen wird angezeigt, wenn Sie `gpg --version` in der Konsole ausführen. So verschlüsselte Dateien können mit jeder GnuPG-Installation und dem vergebenen Passwort wieder entschlüsselt werden. Rufen Sie dafür einfach folgenden Befehl auf:

```
daniel@example:~# gpg --output datei.txt -d datei.txt.gpg
```

**Listing 31.52** PSK entschlüsseln

Ohne die Angabe des Parameters `--output` wird der Inhalt auf der Konsole ausgegeben.

Falls Sie bereits im Besitz eines GnuPG-Schlüssels sind (da Sie diesen eventuell bereits für die sichere E-Mail-Kommunikation verwenden), können Sie diesen auch zum Verschlüsseln von Dateien verwenden:

```
daniel@example:~# gpg -e -r 'um@example.com' datei.txt
```

**Listing 31.53** GnuPG: Schlüssel verwenden

Entsprechend können Sie die Datei auch wieder entschlüsseln:

```
ute@example:~# gpg --output datei.txt -d datei.txt.gpg
```

```
Please enter the passphrase to unlock the OpenPGP secret key:
"Ute Musterfrau <um@example.com>"
3072-bit RSA key, ID 8B77D4FD6E0B1826,
created 2023-01-13 (main key ID 45E6E1066A996F14).
```

```
### HIER FOLGT DIE ABFRAGE DER PASSPHRASE ###
```

```
gpg: encrypted with 3072-bit RSA key, ID 8B77D4FD6E0B1826, created 2023-01-13  
"Ute Musterfrau <um@example.com>"
```

**Listing 31.54** GnuPG: Entschlüsselung

### 31.6.2 Devices

Eine weitere Möglichkeit, um sensible Daten sicher aufzubewahren, besteht darin, diese auf einem verschlüsselten externen Medium aufzubewahren. Das Standardverfahren zur Festplattenverschlüsselung unter Linux ist LUKS (*Linux Unified Key Setup*). Es eignet sich auch für einzelne Partitionen, die ebenfalls auf externen Medien liegen dürfen. Installieren Sie das Paket *cryptsetup* aus den Paketquellen Ihrer Distribution. Schließen Sie nun das externe Device an (USB-Stick oder eine externe Festplatte), und hängen Sie es direkt wieder mittels `umount` aus:

```
daniel@example:~# sudo umount /media/<UID>
```

**Listing 31.55** Aushängen des externen Speichermediums



#### **OpenSUSE: Keine Pakete im Standardrepository!**

Leider ist *cryptsetup* nicht in den Standardpaketquellen von openSUSE enthalten. Installieren Sie daher das Repository von Dirk Müller, um es dennoch zu erhalten:

```
$ sudo zypper addrepo https://download.opensuse.org/repositories/  
home:dirkmuller:Factory/standard/home:dirkmuller:Factory.repo  
$ sudo zypper refresh  
$ sudo zypper install cryptsetup
```

Falls dieses Gerät bereits Daten enthält, sichern Sie diese vorher, da alle Inhalte restlos entfernt werden. Da bei einer klassischen Formatierung immer »Reste« zurückbleiben, sollten Sie das Medium vor der Verschlüsselung vollständig mit Zufallsdaten beschreiben, sodass keine Rückschlüsse auf Ihre alten Daten gezogen werden können. Dies können Sie mit `dd` erreichen:

```
daniel@example:~# sudo dd bs=2M if=/dev/urandom of=/dev/<DEVICE NAME>
```

**Listing 31.56** Medium mit Zufallsdaten überschreiben

Bei großen Medien kann dieser Vorgang Stunden dauern – Zeit, die bei einem eventuellen Verlust aber gut investiert sein kann. Anschließend können Sie das Medium mit *cryptsetup* und *LUKS* verschlüsseln. Führen Sie dazu den nachstehenden Befehl aus:

```
daniel@example:~# sudo cryptsetup luksFormat /dev/<DEVICE NAME>
WARNING!
=====
Daten auf /dev/<DEVICE NAME> werden unwiderruflich überschrieben.

Are you sure? (Type 'yes' in capital letters): YES
Enter passphrase for /dev/<DEVICE NAME>: <PASSWORT>
Verify passphrase: <PASSWORT>
```

**Listing 31.57** Formatierung des Mediums mit »LUKS«

### Richtiges Device wählen

Lassen Sie bei der Auswahl des Device (zum Beispiel `/dev/sdc`) größte Sorgfalt walten. Geben Sie hier zum Beispiel Ihre Homepartition an, wird diese vollständig und unwiderruflich gelöscht.

Falls Sie nicht wissen, welches Device Ihrem externen Medium zugeordnet wurde, können Sie dies mit `dmesg` auslesen.



31

Das externe Medium ist nun von LUKS verschlüsselt und formatiert. Da verschlüsselte Partitionen nicht direkt ins System eingebunden werden können, wird hierfür der *Device Mapper* verwendet. Er stellt eine virtuelle Schicht dar, durch die die entschlüsselte Partition letztendlich eingebunden werden kann. Bevor das Medium eingebunden werden kann, muss aber zuerst ein »lesbares« Filesystem erstellt werden. Binden Sie dafür das verschlüsselte Medium mit `cryptsetup` in Ihr System ein:

```
daniel@example:~# sudo cryptsetup luksOpen /dev/<DEVICE NAME> CRYPT_EXT
Enter LUKS passphrase: <PASSWORT>
```

**Listing 31.58** Öffnen des verschlüsselten Mediums

Der Befehl erwartet zwei Parameter: das eigentliche (verschlüsselte) Medium und die Bezeichnung des entschlüsselten Mediums (im Beispiel `CRYPT_EXT`). Formatieren Sie nun die so eingehängte »entschlüsselte« Partition mit dem von Ihnen bevorzugten Filesystem:

```
daniel@example:~# sudo mkfs.ext4 /dev/mapper/CRYPT_EXT
```

**Listing 31.59** Erstellung des Filesystems im geöffneten Medium

Im Beispiel wurde eine EXT4-Formatierung gewählt. Binden Sie nun das neu angelegte Filesystem in Ihr System ein:

```
daniel@example:~# sudo mount /dev/mapper/CRYPT_EXT /mnt
```

**Listing 31.60** Einhängen des entschlüsselten Mediums

Ihr verschlüsseltes externes Medium ist nun einsatzbereit. Spielen Sie jetzt die eventuell vorher gesicherten Daten zurück, oder legen Sie neue Inhalte an. Abschließend müssen Sie das Filesystem noch aushängen und die »Entschlüsselung« beenden:

```
daniel@example:~# umount /dev/mapper/CRYPT_EXT
daniel@example:~# sudo cryptsetup close CRYPT_EXT
```

#### Listing 31.61 Aushängen des Mediums

Bei Desktop-Betriebssystemen werden Sie nach dem Anschließen des neu verschlüsselten Geräts umgehend nach dem Passwort gefragt. Auf Serversystemen ohne grafische Oberfläche binden Sie das Medium über die Befehle `cryptsetup luksOpen <DEVICE> CRYPT_EXT` und `mount /dev/mapper/CRYPT_EXT /mnt` ein.

### 31.6.3 Festplatten/System

Neben der Verschlüsselung von Devices (USB-Sticks oder Partitionen) ist es ebenso möglich, das ganze System zu verschlüsseln. Dies ist empfehlenswert, denn wenn Sie nur eine Partition des Systems verschlüsseln können zum Beispiel über die Swap-Partition Rückschlüsse auf die Daten gewonnen werden können. So ist ein absoluter Schutz nicht gewährleistet. Es gibt aber auch Nachteile bei einem verschlüsselten System. Eine verschlüsselte Festplatte oder Partition erfordert bei jedem I/O-Zugriff eine Entschlüsselung. Bei kleinen, oft benutzten Dateien ist das kein großes Problem, da diese im Arbeitsspeicher vorgehalten werden. Arbeitet der Benutzer jedoch mit großen Dateien, so kommt es bei einer Verschlüsselung zu Performance-Einbußen. Die Einbuße ist aber zu verkraften, da aktuelle Systeme deutlich schneller rechnen, als Festplatten lesen oder schreiben können.

#### Debian

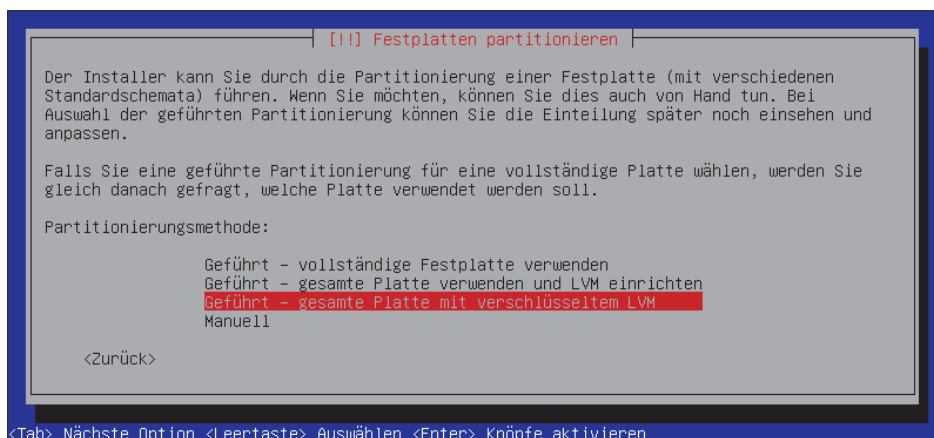
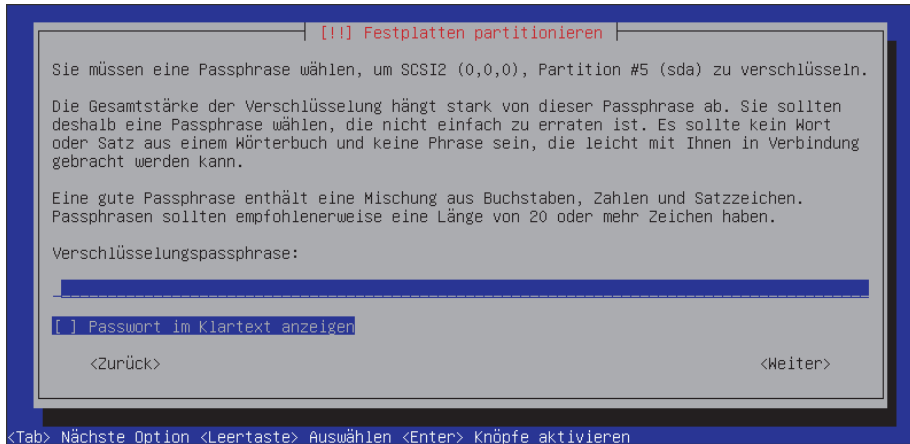


Abbildung 31.5 Debian: Systemverschlüsselung



Unter Debian wird Ihnen beim Auswählen der Partitionierung während der Installation eine vollständige Verschlüsselung angeboten (siehe Abbildung 31.5). Wenn Sie den Punkt GEFÜHRT – GESAMTE PLATTE MIT VERSCHLÜSSELTEM LVM auswählen, werden Sie im weiteren Verlauf nach dem Passwort für die Verschlüsselung gefragt:



31

Abbildung 31.6 Debian: Passwortvergabe

Nach jedem Reboot bleibt das System stehen und verlangt die Eingabe des vergebenen Passworts, damit die Systempartitionen eingebunden werden können:

```
Please unlock disk sda5_crypt: <PASSWORT>
```

Listing 31.62 Debian: Passwordeingabe beim Systemstart

## Ubuntu

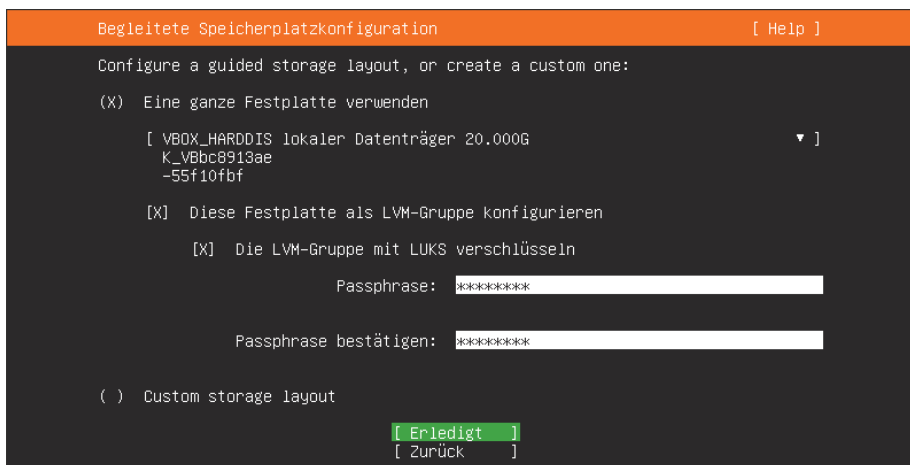


Abbildung 31.7 Ubuntu: Systemverschlüsselung

Unter Ubuntu wird Ihnen beim Auswählen der Partitionierung während der Installation eine vollständige Verschlüsselung angeboten (siehe Abbildung 31.7).

Wenn Sie DIE LVM-GRUPPE MIT LUKS VERSCHLÜSSELN aktivieren und das entsprechende Passwort vergeben, wird die Festplatte verschlüsselt.

## CentOS

Selbstverständlich ist auch CentOS in der Lage, das ganze System zu verschlüsseln. Hierbei gestaltet sich die Installation nur unbedeutend schwieriger. Wählen Sie zunächst den Punkt INSTALLATIONSZIEL. Dort können Sie das Häkchen bei MEINE DATEN VERSCHLÜSSELN setzen (siehe Abbildung 31.8).

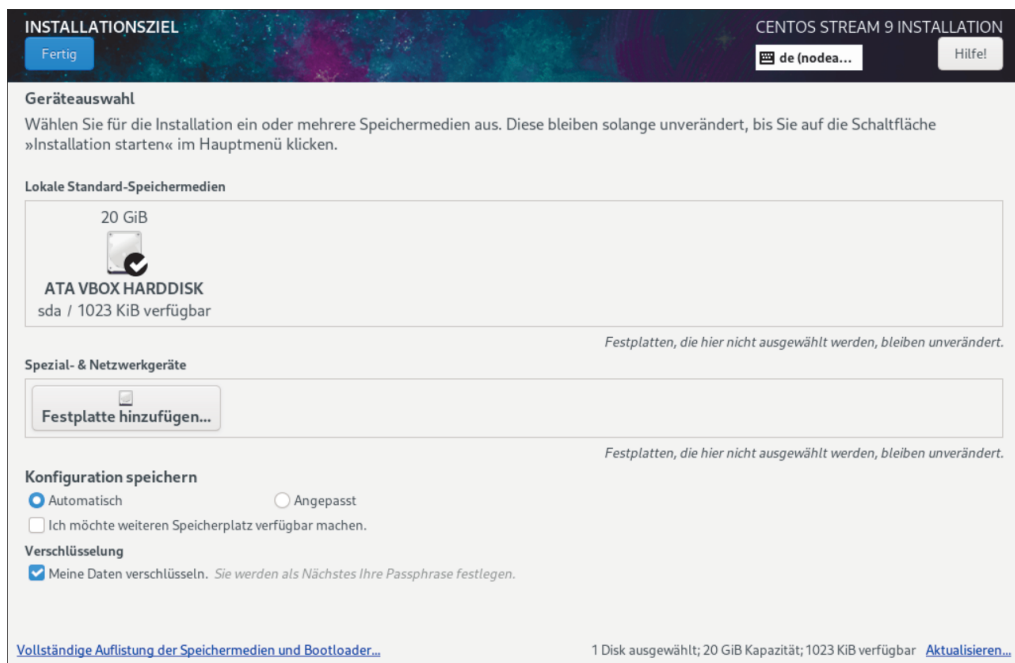


Abbildung 31.8 CentOS: Verschlüsselung aktivieren

Wie im Beschreibungstext angegeben, werden Sie als Nächstes nach dem Passwort für die Verschlüsselung gefragt.

Die restliche Installation verläuft wie üblich. Bei jedem Systemstart werden Sie aufgefordert, das Verschlüsselungspasswort einzugeben.

## openSUSE Leap

Auch mit openSUSE Leap können Sie das ganze System verschlüsseln. Wählen Sie zunächst beim Punkt VORSCHLAG FÜR PARTITIONIERUNG den Unterpunkt GEFÜHRTES SETUP aus. Dann

wählen Sie die Festplatte, setzen das Häkchen bei `ENABLE DISK ENCRYPTION` und vergeben das Festplattenpasswort (siehe Abbildung 31.9).

Auch bei openSUSE Leap verläuft die restliche Installation wie üblich, und Sie werden bei jedem Systemstart aufgefordert, das Festplattenpasswort einzugeben.

**Partitionierungsschema**

Logical Volume Management (LVM) aktivieren

Enable Disk Encryption

Passwort

Passwort Überprüfen

Hilfe    Versionshinweise...    Abbrechen    Zurück    Weiter

Abbildung 31.9 openSUSE Leap: Verschlüsselung einrichten

### Fazit

Auch ein vollständig verschlüsseltes System bietet leider keinen hundertprozentigen Schutz. Wissenschaftlern des *Center for Information Technology Policy* der Universität Princeton ist es gelungen, Festplatten von Geräten zu entschlüsseln, die im Ruhemodus waren.

Bei der Entschlüsselung der Festplatten wird im Betrieb der Schlüssel im RAM abgelegt (hier wurden insbesondere die weitverbreiteten DRAMs genannt), somit kann er auch ausgelesen werden. Auch bei Geräten, deren Stromzufuhr unterbrochen wird, bleiben die Informationen im DRAM mehrere Minuten erhalten. Werden die Speicherchips zusätzlich gekühlt, kann sich die Zeitspanne deutlich erhöhen. Trotz dieser Möglichkeit bietet eine Festplattenverschlüsselung gerade für Außendienstmitarbeiter einen enormen Sicherheitszuwachs.



## Die Autoren



### **Dirk Deimeke (dirk@deimeke.net)**

Dirk Deimeke, Jahrgang 1968, beschäftigt sich seit 1996 aktiv mit Linux. 2008 in die Schweiz ausgewandert, arbeitet er seit April 2022 als »Head of Customer Design & Delivery« für die Swisscom (Schweiz) AG. Dort zählt unter anderem die Betriebsverantwortung einer mittleren dreistelligen Anzahl von Anwendungen und das Engineering von Diensten in der Global Public Cloud zu seinen Aufgaben. Dirk ist an den Podcasts »Buzz-Zoom« und »TILpod« beteiligt, die er auch mitgegründet hat. Er schreibt nicht nur Artikel im eigenen Blog »Dirks Logbuch«, sondern hält auch Vorträge und Workshops auf Open-Source-Konferenzen. Er lebt mit seiner Frau, einem Hund und einem Pferd in einem kleinen Dorf vor den Toren von Zürich.



### **Peer Heinlein (p.heinlein@heinlein-support.de)**

Peer Heinlein, Jahrgang 1976, ist Linuxer der ersten Stunde und E-Mail-Spezialist, dem der Schutz der Privatsphäre und die sichere geschützte Kommunikation sehr am Herzen liegt. Er ist Autor zahlreicher Linux-Fachbücher, zum Beispiel »Postfix« oder »Dovecot«. Mit seinem Team von Heinlein Support berät er Unternehmen und Internet-Provider in Sicherheitsfragen rund um elektronische Kommunikation und betreibt die »Heinlein Akademie«, die in den vergangenen Jahren viele Tausend Administratoren ausgebildet hat. Bereits 1989 gründete Peer Heinlein den Internet Service Provider JPBerlin.de – einen der ersten Anbieter für private E-Mail-Adressen und privates Webhosting in Deutschland. Heute betreibt er mit mailbox.org einen mehrfach als Testsieger ausgezeichneten, ganz besonders auf Verschlüsselung und Datenschutz ausgerichteten internationalen Mailanbieter.



**Stefan Kania (stefan@kania-online.de)**

Stefan Kania, Jahrgang 1961, ist ausgebildeter Informatiker und seit 1997 freiberuflich als Consultant und Trainer tätig. Seine Schwerpunkte liegen in der Implementierung von Samba, Kerberos und LDAP sowie in Schulungen zu den verschiedensten Themen. Seit nunmehr 9 Jahren lebt er in Schleswig-Holstein. Zusammen mit seiner Lebensgefährtin und den beiden Hunden Katie (Border-Terrier) und Paul (Pudel) ist er in Deutschland mit dem Wohnwagen zu den verschiedensten Dogdance-Turnieren unterwegs.



**Axel Miesen (mail@am42.de)**

Axel Miesen, Jahrgang 1970, lernte Linux-Systeme während seines Studiums der Mathematik und Informatik an der Universität Kaiserslautern kennen und schätzen. Nach dem Diplom im Jahr 1998 fand er recht schnell den Weg in die Selbstständigkeit und gibt seitdem sein Wissen und seine Erfahrungen mit Linux- und Open-Source-Themen als Berater und Trainer weiter. In seiner Freizeit macht er gerne mit seinem E-Mountainbike die Wälder unsicher oder versucht, seine Klavier- und Schachkünste zu verbessern. Axel wohnt mit seiner Lebensgefährtin und seiner Tochter in Tauberbischofsheim.



**Daniel van Soest (daniel@vansoest.org)**

Daniel van Soest, Jahrgang 1981, arbeitet beim Kommunalen Rechenzentrum Niederrhein (KRZN) in Kamp-Lintfort; die Themenschwerpunkte seiner Tätigkeit als Senior Consultant liegen in der IT-Security, Compliance und Weiterentwicklung. Neben dem Schreiben von Büchern und Artikeln für Fachzeitschriften lebt er überschüssige Kreativität in seiner Band »4Dirty5« an der E-Gitarre aus. Er wohnt mit seiner Frau und Kindern in Moers an der Schwelle vom Niederrhein zum Ruhrgebiet.

# Index

|                  |     |
|------------------|-----|
| .local .....     | 539 |
| /etc/hosts ..... | 540 |
| 389 DS .....     | 638 |

## A

---

|                                    |      |
|------------------------------------|------|
| access-Einträge .....              | 672  |
| <i>add</i> .....                   | 672  |
| <i>auth</i> .....                  | 672  |
| <i>compare</i> .....               | 672  |
| <i>delete</i> .....                | 672  |
| <i>disclose</i> .....              | 672  |
| <i>manage</i> .....                | 673  |
| <i>none</i> .....                  | 672  |
| <i>read</i> .....                  | 672  |
| <i>search</i> .....                | 672  |
| <i>write</i> .....                 | 672  |
| access-Maps .....                  | 323  |
| ACL, regex .....                   | 679  |
| acme .....                         | 1251 |
| acme.sh (Let's Encrypt) .....      | 1270 |
| acme2certifier .....               | 1264 |
| address_verify .....               | 329  |
| Advanced Routing .....             | 887  |
| alien .....                        | 205  |
| Amanda .....                       | 240  |
| Ansible                            |      |
| <i>Ad-hoc-Kommando</i> .....       | 1084 |
| <i>ansible-console</i> .....       | 1088 |
| <i>ansible-playbook</i> .....      | 1103 |
| <i>ansible.cfg</i> .....           | 1075 |
| <i>console_prompt</i> .....        | 1089 |
| <i>fact_caching</i> .....          | 1123 |
| <i>force_handlers</i> .....        | 1111 |
| <i>retry_files_enabled</i> .....   | 1101 |
| <i>retry_files_save_path</i> ..... | 1101 |
| <i>ansible_become</i> .....        | 1080 |
| <i>ansible_become_method</i> ..... | 1080 |
| <i>ansible_become_pass</i> .....   | 1080 |
| <i>ansible_become_user</i> .....   | 1080 |
| <i>ansible_env</i> .....           | 1137 |
| <i>ansible_password</i> .....      | 1077 |
| <i>ansible_user</i> .....          | 1077 |
| Ausführung                         |      |
| <i>bedingte</i> .....              | 1124 |
| <i>parallele</i> .....             | 1087 |

|                                       |            |
|---------------------------------------|------------|
| <i>Block</i> .....                    | 1134       |
| <i>block</i> .....                    | 1134       |
| <i>changed_when</i> .....             | 1119       |
| <i>Collections</i> .....              | 1137       |
| <i>installieren</i> .....             | 1139       |
| <i>Suchpfad</i> .....                 | 1139       |
| <i>Control Host</i> .....             | 1067       |
| <i>environment</i> .....              | 1136       |
| <i>Extra Vars</i> .....               | 1113       |
| <i>Fact</i> .....                     | 1122       |
| <i>Cache</i> .....                    | 1123       |
| <i>failed_when</i> .....              | 1133       |
| <i>Fehlerbehandlung</i> .....         | 1101, 1133 |
| <i>Filter</i> .....                   | 1127       |
| <i>bool</i> .....                     | 1128       |
| <i>default()</i> .....                | 1128       |
| <i>int</i> .....                      | 1127       |
| <i>join()</i> .....                   | 1127       |
| <i>lower</i> .....                    | 1128       |
| <i>replace()</i> .....                | 1128       |
| <i>upper</i> .....                    | 1128       |
| <i>force_handlers</i> .....           | 1111       |
| <i>Fork</i> .....                     | 1087       |
| <i>gather_facts</i> .....             | 1098, 1122 |
| <i>Geschichte</i> .....               | 1067       |
| <i>group_vars</i> .....               | 1115       |
| <i>Grundkonfiguration</i> .....       | 1075       |
| <i>Grundmerkmale</i> .....            | 1066       |
| <i>Handler</i> .....                  | 1108, 1110 |
| <i>Host-Key-Checking</i> .....        | 1077       |
| <i>host_vars</i> .....                | 1117       |
| <i>Idempotenz</i> .....               | 1086       |
| <i>ignore_errors</i> .....            | 1133       |
| <i>Installation</i> .....             | 1068       |
| <i>Distributionspakete</i> .....      | 1069       |
| <i>PIP</i> .....                      | 1070       |
| <i>Inventory</i> .....                | 1076       |
| <i>Alias</i> .....                    | 1078       |
| <i>Auswahl</i> .....                  | 1082       |
| <i>Gruppe</i> .....                   | 1077       |
| <i>Gruppen-Parametrisierung</i> ..... | 1077       |
| <i>Host-Parametrisierung</i> .....    | 1077       |
| <i>statisches</i> .....               | 1076       |
| <i>Inventory-Datei</i> .....          | 1076       |
| <i>Jinja</i> .....                    | 1125       |
| <i>Anweisung</i> .....                | 1127       |

|                                   |                  |                                        |      |
|-----------------------------------|------------------|----------------------------------------|------|
| <i>Ausdruck</i> .....             | 1126             | <i>wait_for_connection</i> .....       | 1153 |
| <i>Filter</i> .....               | 1127             | <i>yum</i> .....                       | 1143 |
| <i>Kommentar</i> .....            | 1126             | <i>zypper</i> .....                    | 1144 |
| <i>Kernkomponenten</i> .....      | 1066             | <i>notify</i> .....                    | 1110 |
| <i>Konfigurationsdatei</i> .....  | 1075             | <i>PATH</i>                            |      |
| <i>Laborumgebung</i> .....        | 1067             | <i>erweitern</i> .....                 | 1137 |
| <i>Link</i>                       |                  | <i>Patterns</i> .....                  | 1089 |
| <i>anlegen</i> .....              | 1146             | <i>ping</i> .....                      | 1072 |
| <i>Lookup</i> .....               | 1134             | <i>Ping-Test</i> .....                 | 1076 |
| <i>Lookup-Plug-in</i> .....       | 1134             | <i>Play</i> .....                      | 1097 |
| <i>csvfile</i> .....              | 1135             | <i>beenden</i> .....                   | 1100 |
| <i>dig</i> .....                  | 1135             | <i>Play Vars</i> .....                 | 1112 |
| <i>env</i> .....                  | 1135             | <i>Playbook</i> .....                  | 1096 |
| <i>file</i> .....                 | 1135             | <i>Proxy</i> .....                     | 1136 |
| <i>pipe</i> .....                 | 1134             | <i>Rechteerhöhung</i> .....            | 1080 |
| <i>meta-Anweisungen</i> .....     | 1151             | <i>register</i> .....                  | 1120 |
| <i>meta: end_play</i> .....       | 1100             | <i>Retry-Files</i> .....               | 1101 |
| <i>meta: flush_handlers</i> ..... | 1111             | <i>Schleifen</i> .....                 | 1128 |
| <i>mode</i> .....                 | 1145             | <i>set_fact</i> .....                  | 1114 |
| <i>Modul</i>                      |                  | <i>SSH-Public-Key</i> .....            | 1072 |
| <i>apt</i> .....                  | 1143             | <i>Suchpfad</i>                        |      |
| <i>assert</i> .....               | 1153             | <i>erweitern</i> .....                 | 1137 |
| <i>blockinfile</i> .....          | 1147             | <i>Symlink</i>                         |      |
| <i>command</i> .....              | 1084, 1118, 1142 | <i>anlegen</i> .....                   | 1146 |
| <i>copy</i> .....                 | 1085, 1145       | <i>Tag</i> .....                       | 1102 |
| <i>cron</i> .....                 | 1149             | <i>Target Host</i> .....               | 1067 |
| <i>debug</i> .....                | 1151             | <i>Task</i> .....                      | 1099 |
| <i>dnf</i> .....                  | 1143             | <i>Template</i> .....                  | 1125 |
| <i>file</i> .....                 | 1085, 1146       | <i>Testumgebung</i> .....              | 1067 |
| <i>firewalld</i> .....            | 1150             | <i>Umgebungsvariable</i>               |      |
| <i>group</i> .....                | 1150             | <i>ANSIBLE_COLLECTIONS_PATHS</i> ..... | 1139 |
| <i>hostname</i> .....             | 1149             | <i>ANSIBLE_CONFIG</i> .....            | 1075 |
| <i>lineinfile</i> .....           | 1147             | <i>ANSIBLE_INVENTORY</i> .....         | 1083 |
| <i>meta</i> .....                 | 1151             | <i>ANSIBLE_NOCOLOR</i> .....           | 1088 |
| <i>package</i> .....              | 1085, 1144       | <i>auf Zielsystem</i> .....            | 1136 |
| <i>package_facts</i> .....        | 1144             | <i>until</i> .....                     | 1132 |
| <i>pause</i> .....                | 1152             | <i>Variable</i> .....                  | 1112 |
| <i>raw</i> .....                  | 1142             | <i>implizite</i> .....                 | 1124 |
| <i>reboot</i> .....               | 1085, 1150       | <i>Präzedenzen</i> .....               | 1113 |
| <i>replace</i> .....              | 1148             | <i>Präzedenzregeln</i> .....           | 1113 |
| <i>script</i> .....               | 1143             | <i>Variablenstruktur</i>               |      |
| <i>service</i> .....              | 1149             | <i>komplexe</i> .....                  | 1117 |
| <i>service_facts</i> .....        | 1149             | <i>vars</i> .....                      | 1112 |
| <i>set_fact</i> .....             | 1153             | <i>Verbindung</i>                      |      |
| <i>shell</i> .....                | 1085, 1118, 1142 | <i>hängende</i> .....                  | 1088 |
| <i>stat</i> .....                 | 1147             | <i>persistente</i> .....               | 1087 |
| <i>template</i> .....             | 1145             | <i>Versionen</i> .....                 | 1067 |
| <i>unarchive</i> .....            | 1148             | <i>Verzeichnisstruktur</i> .....       | 1074 |
| <i>user</i> .....                 | 1085, 1150       | <i>Virtualenv-Umgebung</i> .....       | 1071 |
| <i>wait_for</i> .....             | 1152             | <i>when</i> .....                      | 1124 |



- 
- with\_dict* ..... 1130
  - with\_items* ..... 1129
  - with\_nested* ..... 1130
  - with\_sequence* ..... 1130
  - with\_together* ..... 1131
  - Wortbedeutung ..... 1066
  - YAML ..... 1090
    - Block-Ausdruck ..... 1094
    - Liste ..... 1092
    - Map ..... 1093
    - NULL-Wert ..... 1095
    - Syntax ..... 1090
    - verschachtelte Strukturen ..... 1093
  - Antivirus (Mail) ..... 348
  - Apache ..... 271
    - Header ..... 302
    - HTTPS anbieten ..... 275
    - ModSecurity ..... 280
    - PHP ..... 294
    - Server Name Indication ..... 278
    - Server-Status ..... 287
    - TLS ..... 302
    - Tuning ..... 285
    - virtuelle Hosts ..... 272
  - Apache Directory Studio ..... 667
  - Apache Subversion ..... 1024
  - APIPA ..... 880
  - AppArmor ..... 543, 596, 663, 1187
    - eigene Profile ..... 1190
    - Statusabfrage ..... 1189
  - apt-cacher-ng
    - Clientkonfiguration ..... 222
    - Import ..... 222
    - Konfiguration ..... 220
  - Architektur ..... 60
  - Archiv in einem anderen Verzeichnis
    - entpacken ..... 241
  - Argon2 ..... 643
  - argon2 ..... 639
  - arp ..... 874, 876
  - AutoCA ..... 632
  - autoca ..... 695
  - AutoYaST ..... 238
  - awk ..... 1044
- B**
- 
- B+-Baum ..... 147
  - B-Baum ..... 147
  - Backscatter ..... 328
  - Backup ..... 237
    - Cold Backup ..... 239
    - differenziell ..... 239
    - Hot Backup ..... 239
    - inkrementell ..... 239
    - loop device ..... 252
    - vollständig ..... 239
    - Wechseldatenträger ..... 246
  - Backupmethode ..... 238
  - Backupstrategie ..... 237, 238
  - Bacula ..... 240
  - Bareos ..... 240
  - bash ..... 1048
    - declare ..... 1056
    - Expansion ..... 1048
    - functions ..... 1056
    - local ..... 1055
    - Logging in Skripten ..... 1057
    - Operationen ..... 1050
    - Spezialparameter ..... 1053
    - test ..... 1057
    - Tipps und Tricks ..... 1060
    - Variablen ..... 1055
  - Bayes-Filter ..... 364
  - Bazaar → GNU Bazaar
  - Berechtigungsstruktur ..... 166
  - Betrieb ..... 61
  - Bind-Mount ..... 77
  - Bind9 ..... 542, 595, 596
  - bind9
    - Ablauf der Namensauflösung ..... 940
    - auto-dnssec ..... 966
    - autoritativ ..... 939
    - DNS over HTTPS (DoH) ..... 967
    - dnscrypt-proxy ..... 971
    - DNSSEC ..... 957
    - dnssec-keygen (KSK) ..... 961
    - dnssec-keygen (ZSK) ..... 960
    - dnssec-signzone ..... 961, 962
    - dsset ..... 965
    - inline-signing ..... 966
    - iterativ ..... 940
    - Key-Signing Key (KSK) ..... 959
    - rekursiv ..... 939
    - RRSIG ..... 958
    - SALT berechnen ..... 962
    - Stamp (DoH) ..... 971
    - Zone-Signing Key (ZSK) ..... 959
  - Blacklist (Mail) ..... 323
  - Blacklisting (in Rspamd) ..... 365

|                                            |            |                                      |          |
|--------------------------------------------|------------|--------------------------------------|----------|
| Blattobjekte .....                         | 632        | chroot .....                         | 1184     |
| Blockgröße .....                           | 149        | <i>FTP</i> .....                     | 1185     |
| bond0 .....                                | 898        | ClamAV .....                         | 348      |
| Bonding .....                              | 878, 896   | ClamAV (für Rspamd) .....            | 360      |
| <i>Betriebsmodi</i> .....                  | 896        | classes.conf .....                   | 740      |
| <i>Bonding-Treiber</i> .....               | 897        | Cluster .....                        | 747      |
| Bootloader .....                           | 69         | <i>Knoten</i> .....                  | 747      |
| btree (Postfix) .....                      | 320        | Cluster Resource Manager .....       | 758      |
| BitFS .....                                | 155        | cn=config .....                      | 675, 710 |
| <i>Defragmentierung</i> .....              | 159        | Common Name .....                    | 632      |
| <i>ext-Dateisysteme konvertieren</i> ..... | 157        | Containervirtualisierung             |          |
| <i>mehrere Devices</i> .....               | 160        | <i>Anonymous Volumes</i> .....       | 839      |
| <i>Snapshots</i> .....                     | 159        | <i>Berechtigungen</i> .....          | 812      |
| <i>Subvolumes</i> .....                    | 158        | <i>Betrieb hinter Proxy</i> .....    | 813      |
| <i>verbrauchter Speicherplatz</i> .....    | 160        | <i>Bind Mounts</i> .....             | 838      |
| <b>C</b>                                   |            |                                      |          |
| CA .....                                   | 695        | <i>Build-Cache umgehen</i> .....     | 845      |
| CA.pl .....                                | 656        | <i>Container</i> .....               | 805, 815 |
| Calamaris .....                            | 455        | <i>limitieren</i> .....              | 828      |
| certbot .....                              | 1251       | <i>löschen</i> .....                 | 818      |
| <i>certonly</i> .....                      | 1252       | <i>mit mehreren Diensten</i> .....   | 851      |
| certbot (Let's Encrypt) .....              | 1270       | <i>Packungsdichte</i> .....          | 831      |
| chain .....                                | 695        | <i>Unterschied zur VM</i> .....      | 806      |
| Checkmk .....                              | 1155       | ctop .....                           | 823      |
| <i>Acknowledgement</i> .....               | 1176       | <i>Dangling Images</i> .....         | 845      |
| <i>Agent</i> .....                         | 1155, 1161 | <i>Datenpersistenz</i> .....         | 836      |
| <i>Dashboard</i> .....                     | 1159       | <i>docker build</i> .....            | 844      |
| <i>Downtime</i> .....                      | 1176       | <i>docker commit</i> .....           | 843      |
| <i>Flapping</i> .....                      | 1164       | <i>Docker Compose</i> .....          | 855      |
| <i>Hard-/Soft-State</i> .....              | 1175       | <i>Autostart</i> .....               | 862      |
| <i>Host-Ordner</i> .....                   | 1165       | <i>systemd-Integration</i> .....     | 862      |
| <i>Installation</i> .....                  | 1155       | <i>docker container create</i> ..... | 818      |
| <i>Instanz</i> .....                       | 1157       | <i>docker container logs</i> .....   | 824      |
| <i>Kontaktgruppen</i> .....                | 1173       | <i>docker container ls</i> .....     | 817      |
| <i>Magic Factor</i> .....                  | 1168       | <i>docker container rm</i> .....     | 818      |
| <i>Netzwerkdienste</i> .....               | 1172       | <i>docker container start</i> .....  | 818      |
| <i>Neuer Host</i> .....                    | 1162       | <i>docker container stats</i> .....  | 828      |
| <i>Notification Period</i> .....           | 1176       | <i>docker cp</i> .....               | 843      |
| <i>Notifications</i> .....                 | 1173       | <i>docker image ls</i> .....         | 817      |
| <i>omd</i> .....                           | 1156       | <i>docker image prune</i> .....      | 846      |
| <i>Regelsätze</i> .....                    | 1164       | <i>docker image pull</i> .....       | 818      |
| <i>Schwellwerte</i> .....                  | 1168       | <i>docker image rm</i> .....         | 818      |
| <i>Server</i> .....                        | 1155       | <i>docker image save/load</i> .....  | 825      |
| <i>Service Discovery</i> .....             | 1170       | <i>docker login</i> .....            | 869      |
| <i>SMS-Alarme</i> .....                    | 1174       | <i>docker logout</i> .....           | 869      |
| <i>Test-Systeme</i> .....                  | 1167       | <i>docker network</i> .....          | 833      |
| <i>Zustände</i> .....                      | 1163       | <i>docker run</i> .....              | 817      |
| <i>Überwachung</i> .....                   | 1160       | <i>docker system df</i> .....        | 831      |
|                                            |            | <i>docker system info</i> .....      | 812, 831 |
|                                            |            | <i>docker system prune</i> .....     | 831      |
|                                            |            | <i>docker top</i> .....              | 822      |

|                                      |          |                                        |          |
|--------------------------------------|----------|----------------------------------------|----------|
| <i>docker volume</i> .....           | 839      | <i>standby</i> .....                   | 759      |
| <i>Dockerfile</i>                    |          | Cronjob .....                          | 606      |
| <i>Best Practices</i> .....          | 854      | CUPS .....                             | 725      |
| <i>Direktiven</i> .....              | 846      | <i>Browsing</i> .....                  | 736      |
| <i>Multistage-Build</i> .....        | 853      | <i>BrowseInterval</i> .....            | 736      |
| <i>Dockerfiles</i> .....             | 844      | <i>BrowseRelay</i> .....               | 736      |
| <i>Entrypoint-Skripte</i> .....      | 850      | <i>classes.conf</i> .....              | 740      |
| <i>Healthchecks</i> .....            | 851      | <i>CUPS-Port</i> .....                 | 727      |
| <i>host-gateway</i> .....            | 835      | <i>cupsd.conf</i> .....                | 726      |
| <i>Image</i> .....                   | 815      | <i>Druckaufträge formatieren</i> ..... | 740      |
| <i>löschen</i> .....                 | 818      | <i>Drucker anlegen</i> .....           | 737      |
| <i>Installation von Docker</i> ..... | 809      | <i>Druckerquotas</i> .....             | 739      |
| <i>Installation von Podman</i> ..... | 811      | <i>Klassen einrichten</i> .....        | 738      |
| <i>Interaktive Container</i> .....   | 820      | <i>Limits</i> .....                    | 734      |
| <i>Konfiguration</i> .....           | 814      | <i>MaxClients</i> .....                | 734      |
| <i>Live Restore</i> .....            | 827      | <i>MaxClientsPerHost</i> .....         | 734      |
| <i>live-restore</i> .....            | 827      | <i>MaxCopies</i> .....                 | 734      |
| <i>Logging</i> .....                 | 824      | <i>MaxJobs</i> .....                   | 735      |
| <i>Named Volumes</i> .....           | 838      | <i>MaxJobsPerUser</i> .....            | 734      |
| <i>Networking</i> .....              | 832      | <i>PreserveJobFiles</i> .....          | 735      |
| <i>Portmapping</i> .....             | 834      | <i>PreserveJobHistory</i> .....        | 735      |
| <i>Private Registry</i> .....        | 862      | <i>TempDir</i> .....                   | 735      |
| <i>Proxy</i> .....                   | 813      | <i>LOCAL</i> .....                     | 733      |
| <i>Prozessverwaltung</i> .....       | 822      | <i>OWNER</i> .....                     | 734      |
| <i>Registry</i> .....                | 815      | <i>Policies</i> .....                  | 731      |
| <i>Repository</i> .....              | 815      | <i>Location-Policies</i> .....         | 731, 732 |
| <i>Restart-Policy</i> .....          | 827      | <i>Operation-Policies</i> .....        | 731, 733 |
| <i>Service-Container</i> .....       | 821      | <i>Printer Instances</i> .....         | 741      |
| <i>Umgebungsvariablen</i> .....      | 823      | <i>printers.conf</i> .....             | 740      |
| <i>User Defined Networks</i> .....   | 833      | <i>SYSTEM</i> .....                    | 734      |
| <i>Versionen</i> .....               | 807      | <i>SystemGroup</i> .....               | 734      |
| control-Einträge .....               | 673      | <i>Web-Frontend</i> .....              | 727      |
| <i>break</i> .....                   | 673      | <i>cupsd.conf</i> .....                | 726, 731 |
| <i>continue</i> .....                | 673      | <i>CVS</i> .....                       | 1021     |
| <i>stop</i> .....                    | 673      |                                        |          |
| <i>Corosync</i> .....                | 755      |                                        |          |
| <i>Authkey</i> .....                 | 757      | <b>D</b>                               |          |
| <i>corosync-keygen</i> .....         | 757      | DANE .....                             | 316      |
| <i>corosync.conf</i> .....           | 755      | DAP-Datenbank .....                    | 630      |
| <i>pcs</i> .....                     | 756      | Dateisystemattribut .....              | 179      |
| <i>corosync-keygen</i> .....         | 757      | <i>chattr</i> .....                    | 179      |
| <i>COW (Copy on Write)</i> .....     | 148      | <i>fstab</i> .....                     | 179      |
| <i>crm</i> .....                     | 758, 761 | <i>NFS</i> .....                       | 179      |
| <i>Colocation-Constraints</i> .....  | 768      | Dateisysteme .....                     | 145      |
| <i>Constraints</i> .....             | 768      | <i>BtrFS</i> .....                     | 155      |
| <i>move</i> .....                    | 765      | <i>Ext2/3</i> .....                    | 149      |
| <i>online</i> .....                  | 760      | <i>FAT</i> .....                       | 145      |
| <i>Order-Constraints</i> .....       | 769      | <i>Journaling</i> .....                | 148      |
| <i>rc_defaults</i> .....             | 766      | <i>Monitoring von</i> .....            | 1168     |
| <i>resource</i> .....                | 764      | <i>ReiserFS</i> .....                  | 152      |

|                                         |          |                                     |          |
|-----------------------------------------|----------|-------------------------------------|----------|
| XFS .....                               | 153      | <i>autoritativ</i> .....            | 939      |
| Dateisystemrecht .....                  | 170      | <i>BIND</i> .....                   | 942      |
| ACL .....                               | 170      | <i>CNAME</i> .....                  | 949      |
| <i>Default-ACL</i> .....                | 173      | <i>DENIC</i> .....                  | 954      |
| <i>getfacl</i> .....                    | 172      | <i>DNS over HTTPS (DoH)</i> .....   | 967      |
| <i>mask</i> .....                       | 175      | <i>dnscrypt-proxy</i> .....         | 971      |
| <i>setfacl</i> .....                    | 172      | <i>DNSSEC</i> .....                 | 957      |
| <i>Attribute</i> .....                  | 179      | <i>dnssec-keygen (KSK)</i> .....    | 961      |
| <i>fstab</i> .....                      | 171      | <i>dnssec-keygen (ZSK)</i> .....    | 960      |
| Dateiverschlüsselung .....              | 1279     | <i>dnssec-signzone</i> .....        | 961, 962 |
| Datenbank .....                         | 371      | <i>dsset</i> .....                  | 965      |
| DCC .....                               | 370      | <i>Expire</i> .....                 | 946      |
| dd .....                                | 243, 250 | <i>FQDN</i> .....                   | 945      |
| <i>Master Boot Record</i> .....         | 251      | <i>Glue-Record</i> .....            | 948      |
| <i>MBR sichern</i> .....                | 251      | <i>hint-Zone</i> .....              | 950      |
| <i>Partitionstabelle sichern</i> .....  | 251      | <i>in-addr.arpa</i> .....           | 952      |
| dd_rescue .....                         | 252      | <i>inline-signing</i> .....         | 966      |
| deb-mirror, Clientkonfiguration .....   | 230      | <i>iterativ</i> .....               | 940      |
| deb-Paket .....                         | 203      | <i>Key-Signing Key (KSK)</i> .....  | 959      |
| <i>aus Repositories</i> .....           | 205      | <i>listen-on</i> .....              | 943      |
| <i>lokal</i> .....                      | 205      | <i>listen-on-v6</i> .....           | 943      |
| debmirror .....                         | 224      | <i>local resolver</i> .....         | 939      |
| <i>Konfiguration</i> .....              | 224      | <i>minimum</i> .....                | 946      |
| delta-syncrepl .....                    | 706      | <i>MX-Record</i> .....              | 948      |
| demoCA .....                            | 657      | <i>named.conf</i> .....             | 942      |
| deref .....                             | 695      | <i>Kommentare</i> .....             | 943      |
| Desktop-Virtualisierung .....           | 776      | <i>Zeilenabschluss</i> .....        | 943      |
| DevOps .....                            | 62       | <i>Nameserver</i> .....             | 939      |
| DHCP .....                              | 934      | <i>Negative Cache TTL</i> .....     | 946      |
| <i>Client</i> .....                     | 934      | <i>Records</i> .....                | 941      |
| <i>default-lease-time</i> .....         | 936      | <i>Refresh</i> .....                | 946      |
| <i>DHCPACK</i> .....                    | 935      | <i>rekursiv</i> .....               | 939      |
| <i>DHCPDISCOVER</i> .....               | 935      | <i>Resource Records</i> .....       | 941      |
| <i>DHCPOFFER</i> .....                  | 935      | <i>Retry</i> .....                  | 946      |
| <i>DHCPREQUEST</i> .....                | 935      | <i>Reverse Lookup</i> .....         | 952      |
| <i>Leases</i> .....                     | 937      | <i>RR</i> .....                     | 941, 947 |
| <i>max-lease-time</i> .....             | 936      | <i>RRSIG</i> .....                  | 958      |
| <i>Relay-Agent</i> .....                | 935      | <i>SALT berechnen</i> .....         | 962      |
| <i>statische Zuweisung</i> .....        | 937      | <i>Secondary-Server</i> .....       | 954      |
| <i>subnet</i> .....                     | 936      | <i>Serial</i> .....                 | 946, 955 |
| Diffie-Hellman .....                    | 334      | <i>SOA</i> .....                    | 945      |
| dig .....                               | 950      | <i>Stamp (DoH)</i> .....            | 971      |
| Disaster Recovery .....                 | 237      | <i>TTL</i> .....                    | 944      |
| DKIM .....                              | 367      | <i>Zone-Signing Key (ZSK)</i> ..... | 959      |
| DN .....                                | 670      | <i>Zonentransfer</i> .....          | 954      |
| DNS .....                               | 939      | <i>DNS over HTTPS (DoH)</i> .....   | 967      |
| <i>A-Record</i> .....                   | 947      | <i>DNS-Server</i> .....             | 539      |
| <i>AAAA-Record</i> .....                | 948      | <i>dns.keytab</i> .....             | 542      |
| <i>Ablauf der Namensauflösung</i> ..... | 940      | <i>DNSBL</i> .....                  | 325      |
| <i>auto-dnssec</i> .....                | 966      | <i>dnscrypt-proxy</i> .....         | 971      |

|                                 |          |
|---------------------------------|----------|
| DNSSEC .....                    | 957      |
| dnstyle .....                   | 670      |
| <i>anonymous</i> .....          | 671      |
| <i>base</i> .....               | 670      |
| <i>children</i> .....           | 671      |
| <i>exact</i> .....              | 670      |
| <i>group</i> .....              | 671      |
| <i>one</i> .....                | 671      |
| <i>self</i> .....               | 671      |
| <i>sub</i> .....                | 671      |
| <i>users</i> .....              | 671      |
| <i>Wildcard</i> .....           | 672      |
| DoH (DNS over HTTPS) .....      | 967      |
| domain component (dc) .....     | 654      |
| Domaincontroller .....          | 537, 538 |
| doveadm .....                   | 338      |
| Dovecot .....                   | 335      |
| <i>auth-socket</i> .....        | 332      |
| <i>hochverfügbar</i> .....      | 342      |
| <i>SASL</i> .....               | 332      |
| Dovecot Director .....          | 343      |
| DRBD .....                      | 749      |
| <i>drbdadm</i> .....            | 752–754  |
| <i>global_common.conf</i> ..... | 750      |
| <i>im Cluster</i> .....         | 769      |
| <i>mydrbd.res</i> .....         | 750      |
| drbdadm .....                   | 752–754  |
| Drucker → CUPS                  |          |
| dump .....                      | 512      |
| dumpe2fs .....                  | 151      |
| dynlist, dyngroup .....         | 690      |

## E

|                              |          |
|------------------------------|----------|
| E-Mail-Verschlüsselung ..... | 1271     |
| e2fsck .....                 | 152      |
| edquota .....                | 186      |
| Empfängervalidierung .....   | 328      |
| Engineering .....            | 61       |
| entryCSN .....               | 697      |
| entryUUID .....              | 697      |
| env .....                    | 1052     |
| Environment .....            | 1052     |
| Etherchannel .....           | 896      |
| Expansion .....              | 1048     |
| Ext2 .....                   | 149      |
| Ext3 .....                   | 149      |
| EXTERNAL .....               | 652, 655 |

## F

|                                            |          |
|--------------------------------------------|----------|
| FAI .....                                  | 238      |
| fail2ban .....                             | 1204     |
| <i>fail2ban-regex</i> .....                | 1209     |
| <i>iptables</i> .....                      | 1208     |
| False Positive .....                       | 349      |
| FAT .....                                  | 145      |
| Fencing .....                              | 773      |
| Festplattenverschlüsselung .....           | 1282     |
| Festplattenzugriffe ermitteln .....        | 1000     |
| Fileserver .....                           | 589      |
| Filesystem-Check .....                     | 252      |
| Forward-Zone .....                         | 549, 595 |
| Forwarder .....                            | 542      |
| frontend .....                             | 675      |
| fstab .....                                | 171      |
| FTP .....                                  | 307      |
| <i>aktives FTP</i> .....                   | 307      |
| <i>Anonymous</i> .....                     | 308      |
| <i>Download-Server</i> .....               | 308      |
| <i>IPv6</i> .....                          | 309      |
| <i>LDAP-Anbindung</i> .....                | 313      |
| <i>passives FTP</i> .....                  | 308      |
| <i>Protokoll</i> .....                     | 307      |
| <i>SSL-Verschlüsselung</i> .....           | 311      |
| <i>vsftpd</i> .....                        | 308      |
| <i>Zugriff auf Homeverzeichnisse</i> ..... | 310      |
| FTP-Server → FTP                           |          |

## G

|                             |          |
|-----------------------------|----------|
| Geräteverschlüsselung ..... | 1280     |
| getent .....                | 666      |
| Git .....                   | 1030     |
| Gitea .....                 | 1037     |
| <i>Installation</i> .....   | 1037     |
| <i>Konfiguration</i> .....  | 1039     |
| Gitolite .....              | 1033     |
| <i>Installation</i> .....   | 1033     |
| <i>Konfiguration</i> .....  | 1035     |
| Gitserver .....             | 1033     |
| GNU Bazaar .....            | 1026     |
| GnuPG .....                 | 1249     |
| GPO .....                   | 567      |
| grep .....                  | 1044     |
| Greylisting .....           | 326      |
| groupOfNames .....          | 672, 678 |
| groupOfUniqueNames .....    | 672, 678 |

- groupOfURLs ..... 672, 692
- GRUB
- chroot* ..... 77
  - Recovery* ..... 76
- GRUB 2 ..... 69
- /etc/default/grub* ..... 75
  - grub.cfg* ..... 70
- Gruppenrichtlinien ..... 567
- samba-tool gpo* ..... 567
  - Verknüpfung* ..... 573
- Gruppenrichtlinieneditor ..... 567
- Gruppenrichtlinienobjekt ..... 568
- Gruppenrichtlinienverwaltung ..... 567, 569
- GSSAPI ..... 528
- GTUBE-Testmail ..... 353
- ## H
- 
- H-Baum ..... 147
- hash (Postfix) ..... 320
- Heimdal-Kerberos ..... 537
- Hochverfügbarkeit ..... 747
- Authkey (Corosync)* ..... 757
  - Colocation-Constraints* ..... 768
  - Constraints* ..... 768
  - Corosync* ..... 748, 755
  - corosync-keygen* ..... 757
  - corosync.conf* ..... 755
  - crm* ..... 758
  - move* ..... 765
  - rc\_defaults* ..... 766
  - resource* ..... 764
- CRM-Shell ..... 761
- crmsh ..... 748
- DRBD ..... 748, 749
- drbd-utils ..... 748
- drbdadm ..... 752–754
- global\_common.conf ..... 750
- hosts ..... 749
- mydrbd.res ..... 750
- Namensauflösung ..... 749
- NTP ..... 749
- OCF ..... 762
- online ..... 760
- Open Cluster Framework ..... 762
- Order-Constraints ..... 769
- Pacemaker ..... 755, 758
- pacemaker ..... 748
- pcs ..... 748, 756, 758
- move* ..... 765
  - rc\_defaults* ..... 766
  - resource* ..... 764
- quorum ..... 760
- standby ..... 759
- Zeitserver ..... 749
- ## I
- 
- IANA ..... 903
- ID-Mapping ..... 556
- IDS/IPS ..... 1193
- ifconfig ..... 874, 880
- Image
- erstellen* ..... 252
  - mounten* ..... 252
- IMAP ..... 335
- inet\_interfaces ..... 317
- initrd ..... 78
- erstellen* ..... 78
  - initramfs.conf* ..... 78
  - mkinitramfs* ..... 78
  - modifizieren* ..... 82
  - RAID* ..... 78
- Intermediate-CA ..... 1253
- Internet Assigned Numbers Authority ..... 903
- iotop ..... 1000
- ip ..... 874, 881
- neighbour add* ..... 884
  - address show* ..... 877
  - IP-Adresse hinzufügen* ..... 881
  - label* ..... 880
  - link* ..... 877
  - set* ..... 878
  - link show* ..... 876
  - neighbour* ..... 876, 883
  - route* ..... 875
  - routing* ..... 885, 888
  - add* ..... 886
  - delete* ..... 887
  - Priorität* ..... 890
  - show* ..... 885
- ipcalc ..... 1007
- IPP ..... 726
- iproute2 ..... 874
- iptables ..... 912
- chains* ..... 912
  - connlimit* ..... 932
  - Default Policy* ..... 914
  - Default-Policy* ..... 921
  - DMZ* ..... 922

- 
- DNAT ..... 930  
 DoS ..... 930  
 Filter ..... 915  
 FORWARD ..... 913  
 ICMP ..... 916, 927  
 INPUT ..... 913  
 limit ..... 929  
 limit-burst ..... 929  
 lo ..... 921  
 Logging ..... 928  
 MASQUERADE ..... 930, 931  
 Module ..... 916  
 Multiport-Modul ..... 921  
 NAT ..... 930  
 OUTPUT ..... 913  
 POSTROUTING ..... 913  
 PREROUTING ..... 912  
 recent ..... 932  
 Regeln löschen ..... 914  
 SNAT ..... 930  
 SPI ..... 918  
 TARGET ..... 914, 917  
 IPv6 ..... 904  
   Adresse ..... 904  
   Adressraum ..... 904  
   anycast ..... 906  
   DAD ..... 908  
   Header-Aufbau ..... 908  
   Header-Länge ..... 908  
   Interface Identifier ..... 905, 910  
   ipv6calc ..... 957  
   link-lokal ..... 908, 911  
   Multicast ..... 906  
   Multihoming ..... 910  
   NAT ..... 904  
   Neighbor Cache ..... 908  
   Neighbor Discovery ..... 907  
   Network Prefix ..... 905  
   Netzmaske ..... 905  
   Notation ..... 904  
   ping6 ..... 911  
   Privacy Extension ..... 910  
   ssh ..... 910  
   unicast ..... 905
- J**
- 
- Join (Active Directory) ..... 581  
 journal-remote.conf ..... 406  
 journal-upload.conf ..... 407
- journalctl ..... 397, 399, 400  
   *journald.conf* ..... 401  
     *auto* ..... 401  
     *persistent* ..... 401  
     *Storage* ..... 401  
     *volatile* ..... 401  
   Kernmeldungen ..... 404  
   PID ..... 403  
   since ..... 402  
   until ..... 402  
 journald ..... 94, 397, 399  
 Journaling ..... 148  
   Ext3 ..... 150
- K**
- 
- kadm5.acl ..... 482  
 KDC-Master ..... 496  
 kdestroy ..... 551  
 keepalived ..... 346  
 Kerberos ..... 471  
   ACL ..... 482  
   *addprinc* ..... 484  
   AS ..... 472  
   Authentication Service ..... 472  
   *delprinc* ..... 493  
   *kadmin* ..... 483, 500  
   *kadmin.local* ..... 483  
   *kdb5\_util* ..... 481  
   *kdb5\_util dump* ..... 503  
   KDC ..... 472  
   KDC-Slave ..... 496  
   *kdc.conf* ..... 475, 477  
   kdestroy ..... 496  
   Key Distribution Center ..... 472  
   keytab ..... 490  
   kinit ..... 486  
   klist ..... 486  
   kprop ..... 496  
   kpropd ..... 502  
   *kpropd.acl* ..... 502  
   krb5.conf ..... 475  
   krb5.keytab ..... 490  
   ktadd ..... 500  
   ktrem ..... 494  
   ktutil ..... 492  
   LDAP  
     GSSAPI ..... 527  
     kadm5.acl ..... 534  
     krb5.keytab ..... 520

- 
- krbSubTrees* ..... 522
  - listprincs* ..... 485
  - PAM* ..... 487, 488, 495
  - pam-config* ..... 488
  - Policies* ..... 504
  - Policy* ..... 518
  - Principal* ..... 472
  - Propagation* ..... 496
  - randkey* ..... 485
  - Realm* ..... 472
  - Replikation* ..... 496
  - SRV-Einträge* ..... 497
  - systemd* ..... 503
  - TGS* ..... 472
  - TGT* ..... 472, 488, 495
  - Ticket* ..... 472
  - Ticket Granting Server* ..... 472
  - Ticket Granting Ticket* ..... 472
  - useradd* ..... 489
  - xinetd* ..... 503
  - Key Version Numbers ..... 491
  - Kickstart ..... 238
  - klist ..... 583
  - kpartx ..... 253
  - krb5.conf ..... 542
  - KVM ..... 781
    - libvirt* ..... 786
    - Live Migration* ..... 802
    - Netzwerk* ..... 783
    - virsh* ..... 789
    - virt-clone* ..... 801
    - virt-install* ..... 794
    - virt-top* ..... 801
    - virt-viewer* ..... 801
    - virtio* ..... 782
    - Virtual Machine Manager* ..... 797
    - VM-Installation* ..... 792
  - KVNO ..... 491
- L**
- 
- LAM ..... 518, 667
  - LDAP ..... 629
    - ACL* ..... 669
      - Aufbau* ..... 670
      - Passwörter* ..... 678
    - Apache* ..... 720
      - Cache-Parameter* ..... 721
      - Module* ..... 721
      - Zugriffsparameter* ..... 722
  - Attribute* ..... 633
  - bdb* ..... 644
  - Datenmodell* ..... 632
  - Distinguished Name* ..... 632
  - DIT* ..... 632
  - dynamische Konfiguration* ..... 643
  - Filter* ..... 687
    - Verknüpfung* ..... 688
  - GSSAPI* ..... 527
  - hdb* ..... 644
  - LDIF* ..... 637
  - Objekte* ..... 632
  - OID* ..... 633, 687
    - registrieren* ..... 634
  - ou* ..... 632
  - Overlay* ..... 690, 694
    - dds* ..... 694
    - refint* ..... 694
    - slapd.conf* ..... 690
    - valsort* ..... 694
  - Replikation* ..... 696
  - Schema* ..... 634
  - Squid* ..... 717
  - squid.conf* ..... 717
  - statische Konfiguration* ..... 643
  - supportedControl* ..... 687
  - supportedExtension* ..... 687
  - Syntaxbeschreibung* ..... 637
  - TLS, Verbindungsaufbau* ..... 659
  - UTF-8* ..... 631
  - LDAP Account Manager ..... 518, 533, 553
  - ldapadd ..... 655
  - ldapi ..... 651
  - ldapmodify ..... 667, 668
  - ldaps ..... 660
  - ldapsearch ..... 669
  - ldapwhoami ..... 530
  - Lebenseinstellung ..... 65
  - Let's Encrypt
    - acme.sh* ..... 1270
    - certbot* ..... 1270
  - letsencrypt ..... 1249
    - acme2certifier* ..... 1264
    - certbot* ..... 1251
    - certonly* ..... 1252
    - DNS-01* ..... 1250, 1271
    - Einschränkungen* ..... 1251
    - Funktionsweise* ..... 1250
    - HTTP-01* ..... 1250
    - TLS-ALPN-01* ..... 1250



|                                      |          |
|--------------------------------------|----------|
| libvirt .....                        | 786      |
| XML-Format .....                     | 787      |
| lightdm .....                        | 587      |
| limits.conf .....                    | 196      |
| Link Aggregation .....               | 896      |
| Linux-Client .....                   | 577      |
| winbind .....                        | 579      |
| LIR .....                            | 903      |
| LMTP .....                           | 319, 339 |
| Load Balancing (squid) .....         | 465      |
| Log-Host .....                       | 405      |
| Logical Volume Manager → LVM         |          |
| Loglevel .....                       | 648, 655 |
| loop device .....                    | 252      |
| lptions .....                        | 740      |
| lpstat .....                         | 740      |
| lsof .....                           | 993      |
| Dateizugriffe .....                  | 994      |
| Netzwerkressourcen .....             | 994      |
| LVM .....                            | 110      |
| Aufbau einer Volume Group .....      | 114      |
| Begriffe .....                       | 112      |
| defekte Festplatte ersetzen .....    | 120      |
| Erweiterung einer Volume Group ..... | 118      |
| Erweiterung eines Volumes .....      | 117      |
| Grundlagen .....                     | 112      |
| Installation .....                   | 113      |
| Kommandos .....                      | 132      |
| logical extent .....                 | 112      |
| logical volume .....                 | 112      |
| lvconvert .....                      | 119      |
| lvcreate .....                       | 115      |
| lvdisplay .....                      | 115      |
| lvextend .....                       | 117      |
| Mirroring .....                      | 125      |
| physical extent .....                | 112      |
| physical volume .....                | 112      |
| pvdisplay .....                      | 114      |
| pvmove .....                         | 120      |
| pvcreate .....                       | 114      |
| Spiegelung .....                     | 125      |
| Spiegelung hinzufügen .....          | 119      |
| Thin Provisioning .....              | 129      |
| vgcreate .....                       | 115      |
| vgdisplay .....                      | 115      |
| vgextend .....                       | 118      |
| vgreduce .....                       | 121      |
| volume group .....                   | 112      |
| LVM-Snapshot .....                   | 239      |

## M

|                                      |      |
|--------------------------------------|------|
| Magic Factor (Checkmk) .....         | 1168 |
| Maildir .....                        | 337  |
| Makefile .....                       | 207  |
| autotools .....                      | 208  |
| MariaDB .....                        | 371  |
| Benutzerkonto anlegen .....          | 372  |
| Indextuning .....                    | 390  |
| Installation .....                   | 371  |
| Point-In-Time-Recovery .....         | 394  |
| Replikation .....                    | 373  |
| Root-Kennwort .....                  | 372  |
| Speichertuning .....                 | 384  |
| Master Boot Record .....             | 69   |
| mdadm .....                          | 102  |
| memberof .....                       | 694  |
| Mercurial .....                      | 1028 |
| Metadatenfeld .....                  | 404  |
| Migrationen .....                    | 266  |
| Milter .....                         | 352  |
| Mirror .....                         | 224  |
| Clientkonfiguration .....            | 230  |
| MIT-Kerberos .....                   | 537  |
| mke2fs .....                         | 149  |
| mkfs.btrfs .....                     | 156  |
| mkfs.reiserfs .....                  | 153  |
| mkfs.xfs .....                       | 153  |
| mkinitramfs .....                    | 78   |
| mkinitrd .....                       | 80   |
| mobile SSH .....                     | 1014 |
| ModSecurity .....                    | 280  |
| mosh .....                           | 1014 |
| Mount Count (Ext) .....              | 150  |
| mtr .....                            | 1006 |
| Multi Provider Replication .....     | 696  |
| Multi-Provider-Replikation .....     | 630  |
| Multi-Provider-Umgebung .....        | 643  |
| Multiprovider-Replikation(MPR) ..... | 706  |
| my traceroute .....                  | 1006 |
| myhostname .....                     | 317  |
| mynetworks .....                     | 318  |
| MySQL .....                          | 371  |

## N

|                  |      |
|------------------|------|
| Nagios .....     | 1155 |
| named.conf ..... | 942  |
| directory .....  | 943  |

|                                          |          |                                  |            |
|------------------------------------------|----------|----------------------------------|------------|
| <i>forwarders</i> .....                  | 944      | <i>lockd</i> .....               | 610        |
| <i>Port</i> .....                        | 943      | <i>mountd</i> .....              | 610        |
| <i>zone</i> .....                        | 944      | <i>Portmapper</i> .....          | 610        |
| named.conf.local .....                   | 542, 596 | <i>Pseudodateisystem</i> .....   | 611        |
| named.conf.options .....                 | 542, 596 | <i>root_squash</i> .....         | 613        |
| Namensstandard .....                     | 163      | <i>rpcinfo</i> .....             | 614        |
| Nameserver                               |          | <i>rsize</i> .....               | 620        |
| <i>Ablauf der Namensauflösung</i> .....  | 940      | <i>secure</i> .....              | 613        |
| <i>auto-dnssec</i> .....                 | 966      | <i>statd</i> .....               | 610, 611   |
| <i>autoritativ</i> .....                 | 939      | <i>subtree-checking</i> .....    | 614        |
| <i>DNS over HTTPS (DoH)</i> .....        | 967      | <i>sync</i> .....                | 613        |
| <i>dnscrypt-proxy</i> .....              | 971      | <i>UTF-8</i> .....               | 610        |
| <i>DNSSEC</i> .....                      | 957      | <i>wdelay</i> .....              | 615        |
| <i>dnssec-keygen (KSK)</i> .....         | 961      | <i>wsize</i> .....               | 620        |
| <i>dnssec-keygen (ZSK)</i> .....         | 960      | nginx .....                      | 289        |
| <i>dnssec-signzone</i> .....             | 961, 962 | <i>Header</i> .....              | 304        |
| <i>dsset</i> .....                       | 965      | <i>HTTPS anbieten</i> .....      | 293        |
| <i>inline-signing</i> .....              | 966      | <i>PHP</i> .....                 | 295        |
| <i>iterativ</i> .....                    | 940      | <i>virtuelle Hosts</i> .....     | 290        |
| <i>Key-Signing Key (KSK)</i> .....       | 959      | NIC Teaming .....                | 896        |
| <i>rekursiv</i> .....                    | 939      | nignx                            |            |
| <i>RRSIG</i> .....                       | 958      | <i>TLS</i> .....                 | 304        |
| <i>SALT berechnen</i> .....              | 962      | nmap .....                       | 987        |
| <i>Stamp (DoH)</i> .....                 | 971      | NS-Record .....                  | 549        |
| <i>Zone-Signing Key (ZSK)</i> .....      | 959      | nscd .....                       | 582        |
| NDS, filelocking .....                   | 609      | nsswitch.conf .....              | 661        |
| NetBIOS-Domainname .....                 | 539      | ntlm .....                       | 586        |
| netlogon .....                           | 550      | ntp .....                        | 552, 601   |
| netstat .....                            | 991      | NTP-Server .....                 | 749        |
| networkd .....                           | 873      | ntp.conf .....                   | 552, 602   |
| Netzwerke, Predictable Network Interface |          | NUD .....                        | 883        |
| Names .....                              | 873      |                                  |            |
| Netzwerkkarte .....                      | 877      | <b>O</b>                         |            |
| <i>Flag</i> .....                        | 877      | OCF .....                        | 762        |
| <i>Interfacename</i> .....               | 879      | olc .....                        | 648        |
| <i>MAC</i> .....                         | 878      | olcAuthzRegexp .....             | 531        |
| <i>MTU</i> .....                         | 878      | olcSpcheckpoint .....            | 697        |
| <i>Netzwerkpräfix</i> .....              | 882      | olcSpssessionlog .....           | 697        |
| <i>primäre IP-Adresse</i> .....          | 882      | OleTools .....                   | 370        |
| Netzwerkmaskenberechnung .....           | 1007     | omd .....                        | 1156, 1157 |
| NFS .....                                | 609      | Open Cluster Framework .....     | 762        |
| <i>async</i> .....                       | 613      | OpenLDAP .....                   | 629        |
| <i>bind-Mountpoint</i> .....             | 611      | OpenSSH .....                    | 973        |
| <i>exportfs</i> .....                    | 615      | <i>passwortloser Login</i> ..... | 978        |
| <i>exports</i> .....                     | 624      | <i>Port-Forwarding</i> .....     | 982        |
| <i>fsid</i> .....                        | 614      | <i>Protokoll</i> .....           | 978        |
| <i>fstab</i> .....                       | 617      | <i>Schlüssel</i> .....           | 978        |
| <i>gss/krb5</i> .....                    | 625      | <i>X11-Forwarding</i> .....      | 981        |
| <i>GSSAPI</i> .....                      | 622      | OpenSSL .....                    | 1253       |
| <i>idmapd</i> .....                      | 617      |                                  |            |

|                                          |          |                                 |          |
|------------------------------------------|----------|---------------------------------|----------|
| CA-Verzeichnisstruktur .....             | 1254     | online .....                    | 760      |
| CRL .....                                | 1262     | Order-Constraints .....         | 769      |
| Intermediate-CA .....                    | 1253     | pcs .....                       | 758      |
| Kommandos .....                          | 1263     | move .....                      | 765      |
| Konfiguration .....                      | 1256     | rc_defaults .....               | 766      |
| openssl.cnf .....                        | 1256     | resource .....                  | 764      |
| widerrufen .....                         | 1262     | quorum .....                    | 760      |
| openssl.cnf .....                        | 656      | Ressource .....                 | 762      |
| openVPN .....                            | 1210     | standby .....                   | 759      |
| Debug .....                              | 1231     | Paketverwaltung .....           | 201      |
| DNS Leak (Windows 10) .....              | 1230     | deb .....                       | 203      |
| easy-rsa .....                           | 1211     | Pakete erzeugen .....           | 210      |
| Clientzertifikat .....                   | 1218     | rpm .....                       | 202      |
| Serverzertifikat .....                   | 1215     | Update-Sicherheit .....         | 217      |
| TLS-Auth .....                           | 1218     | PAM .....                       | 188, 487 |
| easy-rsa init-pki .....                  | 1214     | /etc/pam.d .....                | 194      |
| explicit-exit-notify .....               | 1230     | Konfiguration .....             | 194      |
| Modemverbindungen                        |          | Kontrollflags .....             | 189      |
| (DSL-Modem/UMTS) .....                   | 1231     | Modulargumente .....            | 190      |
| register-dns .....                       | 1229     | Modulaufgaben .....             | 191      |
| Roadwarrior .....                        | 1219     | Modulpfade .....                | 190      |
| Client .....                             | 1223     | pam-config .....                | 488      |
| Server .....                             | 1220     | PAM-Module .....                | 577      |
| Roadwarrior, DNS Leak (Windows 10) ..... | 1230     | pam-mount .....                 | 584      |
| Simple-HA .....                          | 1228     | pam_mount.conf.xml .....        | 585      |
| Site-to-site .....                       | 1226     | Pamifizieren .....              | 188      |
| Tipps und Tricks .....                   | 1229     | parent (squid) .....            | 464      |
| Debug .....                              | 1231     | Partitionierung .....           | 165      |
| explicit-exit-notify .....               | 1230     | Password Settings Objects ..... | 563      |
| register-dns .....                       | 1229     | Passwort .....                  | 559      |
| Windows-Installationspfad .....          | 1230     | Passwortrichtlinien .....       | 562      |
| Windows-Routing/Netzwerk .....           | 1230     | Patchen .....                   | 214      |
| tun/tap .....                            | 1211     | diff .....                      | 214      |
| Zertifikatsverteilung .....              | 1219     | patch .....                     | 216      |
| Operation .....                          | 61       | pcs .....                       | 756, 758 |
| others .....                             | 165      | Colocation-Constraints .....    | 768      |
| otp .....                                | 695      | Constraints .....               | 768      |
| Overlay .....                            | 690      | move .....                      | 765      |
|                                          |          | Order-Constraints .....         | 769      |
|                                          |          | rc_defaults .....               | 766      |
|                                          |          | resource .....                  | 764      |
|                                          |          | standby .....                   | 759      |
|                                          |          | unstandby .....                 | 760      |
|                                          |          | pdbedit .....                   | 553      |
|                                          |          | PdcEmulationMasterRole .....    | 603      |
|                                          |          | Perfect Forward Secrecy .....   | 334      |
|                                          |          | Pflogsumm .....                 | 327      |
|                                          |          | PGP .....                       | 1249     |
|                                          |          | PHP .....                       | 294      |
|                                          |          | Apache .....                    | 294      |
| <b>P</b>                                 |          |                                 |          |
| Pacemaker .....                          | 755, 758 |                                 |          |
| Colocation-Constraints .....             | 768      |                                 |          |
| Constraints .....                        | 768      |                                 |          |
| crm .....                                | 758      |                                 |          |
| move .....                               | 765      |                                 |          |
| rc_defaults .....                        | 766      |                                 |          |
| resource .....                           | 764      |                                 |          |
| CRM-Shell .....                          | 761      |                                 |          |
| OCF .....                                | 762      |                                 |          |

|                                           |          |
|-------------------------------------------|----------|
| <i>nginx</i> .....                        | 295      |
| <i>php-cli</i> .....                      | 299      |
| <i>php-fpm</i> .....                      | 295      |
| <i>Upload-Limit</i> .....                 | 300      |
| Point-In-Time-Recovery .....              | 394      |
| Policy Routing .....                      | 887      |
| <i>PRDB</i> .....                         | 887      |
| <i>Priorität</i> .....                    | 890      |
| POP3 .....                                | 335      |
| Port-Scans .....                          | 987      |
| Postfix .....                             | 315      |
| <i>als Relay</i> .....                    | 319      |
| <i>Grundkonfiguration</i> .....           | 316      |
| <i>main.cf</i> .....                      | 316      |
| <i>master.cf</i> .....                    | 316      |
| <i>Milter</i> .....                       | 352      |
| Postgrey .....                            | 327      |
| postmap .....                             | 321      |
| PostScript .....                          | 725      |
| PPD .....                                 | 731, 743 |
| Predictable Network Interface Names ..... | 873      |
| printers.conf .....                       | 740      |
| proc .....                                | 137      |
| <i>config.gz</i> .....                    | 139      |
| <i>CPU</i> .....                          | 137      |
| <i>net</i> .....                          | 142      |
| <i>RAM</i> .....                          | 138      |
| <i>sysctl.conf</i> .....                  | 143      |
| Provider .....                            | 643      |
| Provision .....                           | 541      |
| Proxy, Squid .....                        | 425      |
| PSO .....                                 | 563      |
| Public Key Infrastructure .....           | 1246     |
| <i>einrichten</i> .....                   | 1253     |
| pulledpork (snort) .....                  | 1201     |
| PuTTY .....                               | 1009     |
| <b>Q</b>                                  |          |
| Quarantäne (E-Mail) .....                 | 348      |
| quorum (HA) .....                         | 760      |
| Quota-System .....                        | 181      |
| <i>aqouta.group</i> .....                 | 183      |
| <i>aqouta.user</i> .....                  | 183      |
| <i>edquota</i> .....                      | 184      |
| <i>fstab</i> .....                        | 182      |
| <i>grpquota</i> .....                     | 182      |
| <i>Hardlimit</i> .....                    | 185      |
| <i>Journaling-Quotas</i> .....            | 183      |
| <i>quotacheck</i> .....                   | 182      |
| <i>quotaon</i> .....                      | 184      |
| <i>repquota</i> .....                     | 187      |
| <i>Softlimit</i> .....                    | 185      |
| <i>usrquota</i> .....                     | 182      |
| <b>R</b>                                  |          |
| RAID .....                                | 97       |
| <i>Level 0</i> .....                      | 98       |
| <i>Level 1</i> .....                      | 98       |
| <i>Level 10</i> .....                     | 99       |
| <i>Level 5</i> .....                      | 98       |
| <i>Level 6</i> .....                      | 99       |
| <i>mdadm</i> .....                        | 102      |
| <i>softraid</i> .....                     | 101      |
| Razor .....                               | 370      |
| RBL .....                                 | 325      |
| RBL (in Rspamd) .....                     | 362      |
| RBL/DNSBL .....                           | 348      |
| Realm .....                               | 538      |
| ReaR .....                                | 255      |
| <i>default.conf</i> .....                 | 265      |
| <i>EXTERNAL</i> .....                     | 258      |
| <i>Installation</i> .....                 | 256      |
| <i>Konfiguration</i> .....                | 256      |
| <i>local.conf</i> .....                   | 256      |
| <i>Migrationen</i> .....                  | 266      |
| <i>NETFS</i> .....                        | 257      |
| <i>P2P</i> .....                          | 266      |
| <i>P2V</i> .....                          | 266      |
| <i>Recovery-Prozess</i> .....             | 263      |
| <i>REQUESTSTORE</i> .....                 | 257      |
| <i>Rescue-Image</i> .....                 | 257      |
| <i>V2P</i> .....                          | 266      |
| <i>V2V</i> .....                          | 266      |
| Redis .....                               | 349      |
| <i>in Rspamd</i> .....                    | 357      |
| Regedit .....                             | 593      |
| Regex .....                               | 1045     |
| Reguläre Ausdrücke .....                  | 1045     |
| <i>Syntax</i> .....                       | 1046     |
| ReiserFS .....                            | 152      |
| reiserfstune .....                        | 153      |
| Relax and Recover .....                   | 255      |
| relay_domains .....                       | 319, 330 |
| Remote Server Administration Tools .....  | 564      |
| Replikation .....                         | 342, 373 |
| <i>Master-Master</i> .....                | 380      |
| <i>Master-Slave</i> .....                 | 373      |
| Repository-Mirror .....                   | 224      |

- 
- repquota ..... 185
  - Rescue-Image (ReaR) ..... 255
  - resolv.conf ..... 939
  - resolvconf ..... 546
  - Restrictions ..... 321
  - Revers-Zone ..... 595
  - Reverse-Zone ..... 549, 594
  - RID ..... 583
  - RIR ..... 903
  - root-Exploit ..... 316
  - root-Shell ..... 44
  - rootdn ..... 650, 653, 676, 677
  - route ..... 874, 875, 885
  - rpm-Paket ..... 202
    - aus Repositories* ..... 205
    - lokal* ..... 205
  - RSAT ..... 564
  - Rspamd ..... 348
    - actions.conf* ..... 359
    - Aktionen* ..... 357
    - Bayes* ..... 364
    - ClamAV* ..... 360
    - Dateianhänge* ..... 366
    - DCC* ..... 370
    - DKIM* ..... 367
    - Mailheader setzen* ..... 361
    - Multimaps* ..... 365
    - OleTools* ..... 370
    - Passwort* ..... 351
    - Razor* ..... 370
    - RBL* ..... 362
    - RegExp* ..... 367
    - rspamc* ..... 360
    - SpamAssassin* ..... 370
    - Upstream-Quellen* ..... 356
    - Webinterface* ..... 351
    - White-/Blacklisting* ..... 365
  - rsync ..... 243, 602, 603
    - dry-run* ..... 606
    - komprimiertes Backup* ..... 248
    - Sichern über das Netz* ..... 244
    - spiegeln* ..... 244
    - ssh* ..... 249
  - Rsync-Daemon ..... 985
  - rsync-Server ..... 603
  - rsyncd ..... 604, 985
  - rsyncd.conf ..... 604
  - Rsyslog ..... 416
    - Filter, ausdrucksbasiert* ..... 417
    - Filter, eigenschaftsbasiert* ..... 416
- ## S
- 
- samba-regedit ..... 593
  - samba-tool ..... 540, 541, 552, 553
    - group add* ..... 555
    - group addmembers* ..... 557
    - group list* ..... 554
    - group listmembers* ..... 555
    - user* ..... 558
  - Sarg ..... 457
  - SASL ..... 331, 652
  - scp ..... 975
  - screen ..... 987
  - Scripting ..... 1043
  - sed ..... 1044
  - serverID ..... 707
  - sfdisk ..... 251
  - sftp ..... 976
  - SGID ..... 168
  - Shell-Expansion ..... 1048
  - sibling (squid) ..... 464
  - simpleSecurityObject ..... 662
  - Single Sign-on ..... 629
  - slap.access ..... 670
  - slapacl ..... 683, 685
  - slapadd ..... 651
  - slapd ..... 648
  - slapd.conf ..... 643, 645
  - smb.conf ..... 541
  - smbclient ..... 549
  - SMS-Alarme (Checkmk) ..... 1174
  - SMTP-Auth ..... 331
  - smtpd\_recipient\_restrictions ..... 321
  - snort ..... 1194
    - pulledpork* ..... 1201
    - snort.conf* ..... 1195
  - SpamAssassin ..... 370
  - Spamfilter ..... 348
  - Spamhaus ..... 325
  - Spamhaus RBL ..... 362
  - Spezialbits ..... 167
    - SGID* ..... 168
    - Sticky-Bit* ..... 169
    - SUID* ..... 168
  - Squid ..... 425
    - acl-Objekte* ..... 433
    - always\_direct* ..... 465
    - basic\_ldap\_auth* ..... 451
    - basic\_ncsa\_auth* ..... 442
    - Beziehungen* ..... 463

|                                           |          |                                              |      |
|-------------------------------------------|----------|----------------------------------------------|------|
| <i>Cache</i> .....                        | 428, 466 | <i>stash</i> .....                           | 482  |
| <i>Cache anlegen</i> .....                | 470      | <i>stash-file</i> .....                      | 481  |
| <i>cache_dir</i> .....                    | 467      | <i>Sticky-Bit</i> .....                      | 169  |
| <i>cache_mem</i> .....                    | 467      | <i>STONITH (HA)</i> .....                    | 773  |
| <i>cache_peer</i> .....                   | 463      | <i>strace</i> .....                          | 1001 |
| <i>Calamaris</i> .....                    | 455      | <i>Subnetzberechnung</i> .....               | 1007 |
| <i>delay_access</i> .....                 | 462      | <i>Subversion</i> → <i>Apache Subversion</i> |      |
| <i>delay_class</i> .....                  | 460      | <i>suffix</i> .....                          | 645  |
| <i>delay_parameters</i> .....             | 461      | <i>SUID</i> .....                            | 168  |
| <i>delay_pool</i> .....                   | 459      | <i>supportedFeatures</i> .....               | 687  |
| <i>delay_pools N</i> .....                | 460      | <i>SVN</i> → <i>Apache Subversion</i>        |      |
| <i>ext_ldap_group_acl</i> .....           | 454      | <i>syncprov</i> .....                        | 709  |
| <i>ext_wbinfo_group_acl</i> .....         | 453      | <i>syncrepl</i>                              |      |
| <i>Festplatten-Cache</i> .....            | 467      | <i>Consumer</i> .....                        | 696  |
| <i>Festplatten-Cache-Berechnung</i> ..... | 469      | <i>delta-syncrepl</i> .....                  | 697  |
| <i>Hauptspeicher-Cache</i> .....          | 467      | <i>entryCSN</i> .....                        | 696  |
| <i>http_access-Regeln</i> .....           | 436      | <i>entryUUID</i> .....                       | 696  |
| <i>Inodes</i> .....                       | 468      | <i>Provider</i> .....                        | 696  |
| <i>intercept</i> .....                    | 457      | <i>refreshAndPersist</i> .....               | 696  |
| <i>IP-Authentifizierung</i> .....         | 438      | <i>refreshOnly</i> .....                     | 696  |
| <i>Load Balancing</i> .....               | 465      | <i>Session Cookie</i> .....                  | 696  |
| <i>Logging</i> .....                      | 430      | <i>Synergy</i> .....                         | 1013 |
| <i>negotiate_kerberos_auth</i> .....      | 447      | <i>Syslog</i> .....                          | 397  |
| <i>Netzwerk</i> .....                     | 428      | <i>Datenbank</i> .....                       | 420  |
| <i>ntlm_auth</i> .....                    | 445      | <i>Facility</i> .....                        | 397  |
| <i>parent</i> .....                       | 464      | <i>Log-Level</i> .....                       | 397  |
| <i>Sarg</i> .....                         | 457      | <i>Priority</i> .....                        | 397  |
| <i>sibling</i> .....                      | 464      | <i>Remote-Logging</i> .....                  | 418  |
| <i>Testumfeld</i> .....                   | 428      | <i>Severity</i> .....                        | 397  |
| <i>transparent</i> .....                  | 457      | <i>Syslog-ng</i> .....                       | 410  |
| <i>Verzeichnisdienste</i> .....           | 444      | <i>destination</i> .....                     | 410  |
| <i>Verzögerung des Datenverkehrs</i>      |          | <i>Destination-Objekt</i> .....              | 412  |
| ( <i>delay_pools</i> ) .....              | 459      | <i>filter</i> .....                          | 410  |
| <i>ss</i> .....                           | 604      | <i>Filter-Objekt</i> .....                   | 414  |
| <i>SSH</i> .....                          | 973      | <i>log</i> .....                             | 410  |
| <i>ssh (Client)</i> .....                 | 974      | <i>Log-Objekt</i> .....                      | 416  |
| <i>SSH-Agent</i> .....                    | 980      | <i>options</i> .....                         | 410  |
| <i>sshd (Server)</i> .....                | 976      | <i>source</i> .....                          | 410  |
| <i>SshFS</i> .....                        | 983      | <i>Source-Objekt</i> .....                   | 412  |
| <i>SSL-Zertifikat</i>                     |          | <i>template</i> .....                        | 410  |
| <i>für Apache</i> .....                   | 275      | <i>SyslogD</i> .....                         | 408  |
| <i>für Dovecot</i> .....                  | 340      | <i>Systemadministrator</i> .....             | 47   |
| <i>für nginx</i> .....                    | 293      | <i>Arbeitstechniken</i> .....                | 55   |
| <i>für Postfix</i> .....                  | 333      | <i>Aufgaben</i> .....                        | 47   |
| <i>sssd</i> .....                         | 660, 661 | <i>Chief Information Officer</i> .....       | 54   |
| <i>sssd-tools</i> .....                   | 661      | <i>DevOps</i> .....                          | 62   |
| <i>sssd.conf</i> .....                    | 663      | <i>Einordnung</i> .....                      | 60   |
| <i>Standarddrucker</i> .....              | 740      | <i>Fähigkeiten und Fertigkeiten</i> .....    | 54   |
| <i>Standby-Provider-Replication</i> ..... | 696      | <i>Intermediate/Advanced System</i>          |      |
| <i>startTLS</i> .....                     | 660      | <i>Administrator</i> .....                   | 50   |

- 
- IT Director* ..... 53  
*Job-Definitionen* ..... 48  
*Junior System Administrator* ..... 50  
*Management-Level* ..... 52  
*Novice System Administrator* ..... 49  
*Senior System Administrator* ..... 51  
*Softskills* ..... 54  
*System Administration Manager* ..... 53  
*Technical Lead* ..... 53  
*Unternehmensgröße* ..... 49  
*Verhaltenskodex* ..... 64  
*Verhältnis zu anderen Administratoren* ..... 58  
*Verhältnis zu Benutzern* ..... 58  
*Verhältnis zu Chef/Vorgesetzten* ..... 57  
systemd ..... 83, 397, 399, 603, 640  
  *Abhängigkeiten* ..... 92  
  *Begriffe* ..... 84  
  *list-unit-files* ..... 91  
  *list-units* ..... 92  
  *Service Units*  
    *aktivieren und deaktivieren* ..... 87  
    *erstellen* ..... 88  
    *Konfiguration anzeigen* ..... 88  
    *kontrollieren* ..... 85  
  *Target Units* ..... 90  
systemd-journal-remote.service ..... 405  
systemd-journal-remote.socket ..... 405  
SysVinit ..... 83  
sysvol ..... 550, 602
- T**
- 
- tar ..... 241  
  *Archiv erstellen* ..... 241  
  *Archiv im aktuellen Verzeichnis entpacken* ..... 241  
  *Archiv komprimieren* ..... 241  
  *Archiv prüfen* ..... 242  
  *Exkludieren von Dateien* ..... 242  
  *ssh* ..... 243  
  *umask* ..... 243  
tc ..... 874  
Terminal-Multiplexer ..... 987  
Ticket ..... 471  
Ticket-Cache ..... 473  
TLS ..... 629, 656  
  *ldap.conf* ..... 659  
  *ldapsearch* ..... 659  
  *slapd.conf* ..... 658  
top ..... 997  
transport\_maps ..... 319  
tree ..... 165, 167  
Trunking ..... 896  
tune2fs ..... 150
- U**
- 
- udev ..... 133  
ulimit ..... 195  
  *Hardlimit* ..... 197  
  *limits.conf* ..... 196  
  *Softlimit* ..... 197  
umask ..... 169  
Umgebungsvariablen ..... 1052  
update-initramfs ..... 78
- V**
- 
- VCS → Versionskontrollsysteme  
Verbindungsaufbau ..... 549  
Verschlüsselung ..... 1241  
  *acme2certifier* ..... 1264  
  *asymmetrisches Verfahren* ..... 1244  
  *Dateien* ..... 1279  
  *E-Mails* ..... 1271  
    *GnuPG* ..... 1271  
    *S/MIME* ..... 1278  
  *Festplatten* ..... 1282  
  *Geräte* ..... 1280  
  *Historie* ..... 1241  
  *Kerckhoffs Grundsätze* ..... 1243  
  *Public Key Infrastructure* ..... 1246  
    *einrichten* ..... 1253  
  *symmetrisches Verfahren* ..... 1243  
  *Web-of-Trust (PGP)* ..... 1249  
  *X.509-Zertifikate* ..... 1247  
Versionskontrolle ..... 1017  
  *dezentral* ..... 1020  
  *lokal* ..... 1018  
  *Philosophien* ..... 1018  
  *verteilte Systeme* ..... 1020  
  *zentral* ..... 1019  
Versionskontrollsysteme ..... 1017  
  *Apache Subversion* ..... 1024  
  *CVS* ..... 1021  
  *Git* ..... 1030  
  *GNU Bazaar* ..... 1026  
  *Mercurial* ..... 1028  
  *Überblick* ..... 1020

|                                            |      |                                            |               |
|--------------------------------------------|------|--------------------------------------------|---------------|
| Verzeichnisdienst .....                    | 629  | <i>WireGuard</i> .....                     | 1232          |
| Verzeichnisstruktur .....                  | 167  | VRRP .....                                 | 346           |
| Viren                                      |      | vsftpd .....                               | 1185          |
| <i>GTUBE-Pattern</i> .....                 | 353  |                                            |               |
| <i>Makroviren</i> .....                    | 360  | <b>W</b>                                   |               |
| virsh .....                                | 789  |                                            |               |
| virt-clone .....                           | 801  | wbinfo -g .....                            | 582           |
| virt-install .....                         | 794  | wbinfo -u .....                            | 582           |
| virt-top .....                             | 801  | Web-of-Trust .....                         | 1249          |
| virt-viewer .....                          | 801  | Webserver .....                            | 271           |
| Virtual Machine Manager .....              | 797  | <i>Apache</i> .....                        | 271           |
| <i>Live Migration</i> .....                | 802  | <i>nginx</i> .....                         | 289           |
| Virtual Private Network                    |      | what-Feld .....                            | 670           |
| <i>openVPN</i> .....                       | 1210 | Whitelist (Mail) .....                     | 323           |
| <i>WireGuard</i> .....                     | 1232 | Whitelisting (in Rspamd) .....             | 365           |
| Virtual-Maps .....                         | 330  | who-Feld .....                             | 671           |
| VirtualBox .....                           | 776  | winbind .....                              | 556, 577, 581 |
| <i>Import/Export</i> .....                 | 780  | winbindd .....                             | 556           |
| <i>Installationsmedium anlegen</i> .....   | 777  | Windows Remote Server Administration Tools |               |
| <i>Netzwerkkonfiguration</i> .....         | 778  | (RSAT) .....                               | 553, 564      |
| Virtualisierung .....                      | 775  | WinSCP .....                               | 1012          |
| <i>Desktop-Virtualisierung</i> .....       | 776  | WireGuard .....                            | 1232          |
| <i>Grundlagen</i> .....                    | 775  | <i>wg</i> .....                            | 1233          |
| KVM .....                                  | 781  | <i>wg-quick</i> .....                      | 1235          |
| <i>libvirt</i> .....                       | 786  |                                            |               |
| <i>Netzwerk</i> .....                      | 783  | <b>X</b>                                   |               |
| <i>virsh</i> .....                         | 789  |                                            |               |
| <i>virt-install</i> .....                  | 794  | X.500 .....                                | 630           |
| <i>virtio</i> .....                        | 782  | X.509-Zertifikate .....                    | 1247          |
| <i>Live Migration</i> .....                | 802  | Xen .....                                  | 783           |
| <i>Server-Virtualisierung</i> .....        | 780  | XFS .....                                  | 153           |
| <i>vir-clone</i> .....                     | 801  | xinetd .....                               | 603, 605      |
| <i>vir-top</i> .....                       | 801  | xinetd.d .....                             | 603           |
| <i>virt-viewer</i> .....                   | 801  |                                            |               |
| <i>Virtual Machine Manager (VMM)</i> ..... | 797  | <b>Z</b>                                   |               |
| <i>Xen</i> .....                           | 783  |                                            |               |
| vmail (Userkennung) .....                  | 336  | Zeitserver .....                           | 552, 601, 749 |
| VPN                                        |      | Zertifikate .....                          | 1247          |
| <i>openVPN</i> .....                       | 1210 | Zugriffsrechte .....                       | 163           |





## Die Serviceseiten

Im Folgenden finden Sie Hinweise, wie Sie Kontakt zu uns aufnehmen können.

### Lob und Tadel

Wir hoffen sehr, dass Ihnen dieses Buch gefallen hat. Wenn Sie zufrieden waren, empfehlen Sie das Buch bitte weiter. Wenn Sie meinen, es gebe doch etwas zu verbessern, schreiben Sie direkt an den Lektor dieses Buches: *christoph.meister@rheinwerk-verlag.de*. Wir freuen uns über jeden Verbesserungsvorschlag, aber über ein Lob freuen wir uns natürlich auch!

Auch auf unserer Webkatalogseite zu diesem Buch haben Sie die Möglichkeit, Ihr Feedback an uns zu senden oder Ihre Leseerfahrung per Facebook, Twitter oder E-Mail mit anderen zu teilen. Folgen Sie einfach diesem Link: *https://www.rheinwerk-verlag.de/5734*.

### Zusatzmaterialien

Falls Zusatzmaterialien (Beispielcode, Übungsmaterial, Listen usw.) für dieses Buch verfügbar sind, finden Sie sie in Ihrer Online-Bibliothek sowie auf der Webkatalogseite zu diesem Buch: *https://www.rheinwerk-verlag.de/5734*. Wenn uns sinnentstellende Tippfehler oder inhaltliche Mängel bekannt werden, stellen wir Ihnen dort auch eine Liste mit Korrekturen zur Verfügung.

### Technische Probleme

Im Falle von technischen Schwierigkeiten mit dem E-Book oder Ihrem E-Book-Konto beim Rheinwerk Verlag steht Ihnen gerne unser Leserservice zur Verfügung: *ebooks@rheinwerk-verlag.de*.

## Über uns und unser Programm

Informationen zu unserem Verlag und weitere Kontaktmöglichkeiten bieten wir Ihnen auf unserer Verlagswebsite <https://www.rheinwerk-verlag.de>. Dort können Sie sich auch umfassend und aus erster Hand über unser aktuelles Verlagsprogramm informieren und alle unsere Bücher und E-Books schnell und komfortabel bestellen. Alle Buchbestellungen sind für Sie versandkostenfrei.

# Rechtliche Hinweise

In diesem Abschnitt finden Sie die ausführlichen und rechtlich verbindlichen Nutzungsbedingungen für dieses E-Book.

## Copyright-Vermerk

Das vorliegende Werk ist in all seinen Teilen urheberrechtlich geschützt. Alle Nutzungs- und Verwertungsrechte liegen bei den Autor\*innen und beim Rheinwerk Verlag, insbesondere das Recht der Vervielfältigung und Verbreitung, sei es in gedruckter oder in elektronischer Form.

© Rheinwerk Verlag GmbH, Bonn 2023

## Nutzungs- und Verwertungsrechte

Sie sind berechtigt, dieses E-Book ausschließlich für persönliche Zwecke zu nutzen. Insbesondere sind Sie berechtigt, das E-Book für Ihren eigenen Gebrauch auszudrucken oder eine Kopie herzustellen, sofern Sie diese Kopie auf einem von Ihnen alleine und persönlich genutzten Endgerät speichern. Zu anderen oder weitergehenden Nutzungen und Verwertungen sind Sie nicht berechtigt.

So ist es insbesondere unzulässig, eine elektronische oder gedruckte Kopie an Dritte weiterzugeben. Unzulässig und nicht erlaubt ist des Weiteren, das E-Book im Internet, in Intranets oder auf andere Weise zu verbreiten oder Dritten zur Verfügung zu stellen. Eine öffentliche Wiedergabe oder sonstige Weiterveröffentlichung und jegliche den persönlichen Gebrauch übersteigende Vervielfältigung des E-Books ist ausdrücklich untersagt. Das vorstehend Gesagte gilt nicht nur für das E-Book insgesamt, sondern auch für seine Teile (z. B. Grafiken, Fotos, Tabellen, Textabschnitte).

Urheberrechtsvermerke, Markenzeichen und andere Rechtsvorbehalte dürfen aus dem E-Book nicht entfernt werden, auch nicht das digitale Wasserzeichen.

Die automatisierte Analyse des Werkes, um daraus Informationen insbesondere über Muster, Trends und Korrelationen gemäß §44b UrhG (»Text und Data Mining«) zu gewinnen, ist untersagt.

## Digitales Wasserzeichen

Dieses E-Book-Exemplar ist mit einem **digitalen Wasserzeichen** versehen, einem Vermerk, der kenntlich macht, welche Person dieses Exemplar nutzen darf. Wenn Sie diese Person nicht sind, liegt ein Verstoß gegen das Urheberrecht vor, und wir bitten Sie freundlich, das E-Book nicht weiter zu nutzen und uns diesen Verstoß zu melden. Eine kurze E-Mail an [service@rheinwerk-verlag.de](mailto:service@rheinwerk-verlag.de) reicht schon. Vielen Dank!

## Markenschutz

Die in diesem Werk wiedergegebenen Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. können auch ohne besondere Kennzeichnung Marken sein und als solche den gesetzlichen Bestimmungen unterliegen.

## Haftungsausschluss

Ungeachtet der Sorgfalt, die auf die Erstellung von Text, Abbildungen und Programmen verwendet wurde, können weder Verlag noch Autor\*innen, Herausgeber\*innen oder Übersetzer\*innen für mögliche Fehler und deren Folgen eine juristische Verantwortung oder irgendeine Haftung übernehmen.