PEARSON Education

ADDISON-WESLEY

open source library

Michael Gutmann
Detlef Lannert

Linux im Netzwerk

Der Praxisleitfaden für kleine und mittlere Umgebungen

ADDISON-WESLEY

Linux im Netzwerk

open source library

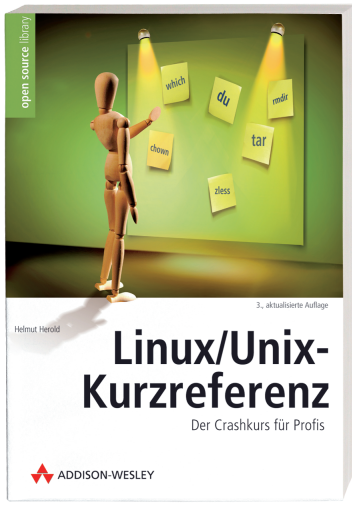
open source library

Open Source Software wird gegenüber kommerziellen Lösungen immer wichtiger. Addison-Wesley trägt dieser Entwicklung Rechnung mit den Büchern der **Open Source Library**. Administratoren, Entwickler und User erhalten hier professionelles Know-how, um freie Software effizient einzusetzen. Behandelt werden sowohl Themen wie Betriebssysteme, Netzwerke und Sicherheit als auch Programmierung.

Eine Auswahl aus unserem Programm:

Die beiden XAMPP-Gründer und -Entwickler Kai Seidler und Kay Vogelgesang beschreiben alle Aspekte der fortgeschrittenen Nutzung von XAMPP – von der Zusammenarbeit mit verschiedenen Datenbanken über die Nutzung der XAMPP-eigenen FTP- und E-Mail-Dienste bis hin zu Performance-Tuning und Sicherheitsfragen. Erfahrene XAMPP-Nutzer lernen außerdem, wie sie die einzelnen Komponenten erweitern und wie sie die bekanntesten PHP-Applikationen unter XAMPP nutzen. Hoher Praxisnutzen ist garantiert: Die beiden Autoren orientieren sich eng an den Fragen und Problemen der Anwender in den XAMPP-Foren.

Das XAMPP-Handbuch
Kai Seidler, Kay Vogelgesang
528 S.
Euro 39,95 (D), 41,10 (A)
ISBN 978-3-8273-2281-4



Das praktische Nachschlagewerk für die tägliche Arbeit mit Linux und/oder Unix in aktualisierter Neuauflage. Diese Kurzreferenz enthält alle Kommandos in übersichtlicher Form zum Nachschlagen für die tägliche Arbeit mit Linux/Unix. Folgende Themen werden behandelt: Linux-Unix-Grundlagen Linux-Unix-Shells awk und sed, lex und yacc, make Netzwerk- und Threadprogrammierung

Linux/Unix-Kurzreferenz
Helmut Herold
360 S.
Euro 19,95 (D), 20,60 (A)
ISBN 978-3-8273-2494-8

PEARSON Education

ADDISON-WESLEY

open source library

Michael Gutmann, Detlef Lannert

Linux im Netzwerk

Der Praxisleitfaden für kleine und mittlere Umgebungen

eBook

Die nicht autorisierte Weitergabe dieses eBooks an Dritte ist eine Verletzung des Urheberrechts!

ADDISON-WESLEY

An imprint of Pearson Education

München • Boston • San Francisco • Harlow, England
Don Mills, Ontario • Sydney • Mexico City
Madrid • Amsterdam

Die Deutsche Bibliothek – CIP-Einheitsaufnahme

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

Die Informationen in diesem Produkt werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht.
Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt.
Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen.
Trotzdem können Fehler nicht ausgeschlossen werden.
Verlag, Herausgeber und Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.
Für Verbesserungsvorschläge und Hinweise auf Fehler sind Verlag und Herausgeber dankbar.

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe und der Speicherung in elektronischen Medien.
Die gewerbliche Nutzung der in diesem Produkt gezeigten Modelle und Arbeiten ist nicht zulässig.

Fast alle Hardware- und Softwarebezeichnungen und weitere Stichworte
und sonstige Angaben, die in diesem Buch verwendet werden,
sind als eingetragene Marken geschützt.
Da es nicht möglich ist, in allen Fällen zeitnah zu ermitteln,
ob ein Markenschutz besteht, wird das ® Symbol in diesem Buch nicht verwendet.

Umwelthinweis:
Dieses Produkt wurde auf chlorfrei gebleichtem Papier gedruckt.

10 9 8 7 6 5 4 3 2 1

08 07

ISBN 978-3-8273-2173-2

© 2007 by Addison-Wesley Verlag,
ein Imprint der Pearson Education Deutschland GmbH,
Martin-Kollar-Straße 10–12, D-81829 München/Germany
Alle Rechte vorbehalten
Einbandgestaltung: Marco Lindenbeck, webwo GmbH (mlindenbeck@webwo.de)
Lektorat: Boris Karnikowski, bkarnikowski@pearson.de
Fachlektorat: Wilhelm Dolle, Berlin
Herstellung: Monika Weiher, mweiher@pearson.de
Satz: LE-TEX Jelonek, Schmidt & Vöckler GbR, Leipzig
Druck und Verarbeitung: Bercker Graph. Betrieb, Kevelaer
Printed in Germany

Inhaltsübersicht

Inhaltsverzeichnis

1 Über dieses Buch

1.1 Für wen ist dieses Buch?

Linux ist im Serverbereich bereits ein alltäglicher Anblick. Wenn Sie einen Blick auf die Hosting-Angebote einschlägiger Computerzeitschriften werfen, finden Sie dort ein großes Angebot an virtuellen oder dedizierten Linux-Servern, mit denen Sie eine Vielzahl von Internet-Services anbieten können. In kleinen Büroumgebungen oder zu Hause, wo Linux noch nicht die Verbreitung hat wie in anderen Bereichen will man oder frau sich die Vorteile eines Netzwerks mit günstigem und leistungsfähigem Server erschließen. Aber auch beim Einsatz von Linux als Desktopsystem werden Ihnen Teile des Buches helfen, besser zu verstehen, was »unter der Haube« passiert.

1.2 Was wollen wir Ihnen vermitteln?

Der Titel dieses Buches ist ein wenig irreführend; wenn wir ehrlich sind, *ist* Linux das Netzwerk, bzw. ist »Netzwerken« ein elementarer Bestandteil von Linux. Sie werden verstehen, dass wir diesen Themenbereich auf 350 Seiten nicht erschöpfend behandeln können. Das ist eigentlich aber auch nicht notwendig, da zu jeder Software, die wir behandeln, eine Online-Dokumentation existiert. Diese hat aber meist Referenzcharakter, was den Einstieg nicht unbedingt erleichtert. Daher wollen wir Ihnen aus unserem Erfahrungsschatz heraus die grundlegenden Konzepte nahebringen und das Wissen vermitteln, das Sie benötigen, um sich nach der Lektüre der jeweiligen Kapitel mit den vorhandenen (Online-)Dokumentationen zurechtzufinden.

1.3 Was sollten Sie schon wissen?

Aber wir wollen ehrlich sein: Dies ist kein Anfängerbuch. Sie sollten entweder bereits grundlegende Erfahrung mit Linux besitzen oder mit anderen Betriebssystemen Erfahrungen beim Aufbau oder Betrieb eines Netzwerks gewonnen haben. Wir werden zwar alle Themen von Anfang an erläutern, aber vielleicht nicht in der Ausführlichkeit, die Ihnen als Linux-Anfänger ein Verständnis erlauben würde. Insbesondere grundlegende Konzepte wie die Ordnerstruktur von Linux, die Konfiguration des Systems bezüglich Grafik oder USB-Hardware, die nichts mit Netz-

werken zu tun hat, wollen wir hier nicht näher erläutern; daher könnte es helfen, ein weiteres, einführendes Linux-Werk zur Hand zu haben.

1.4 Warum so viel Kommandozeile?

Außerdem gehen wir von Szenarien aus, in denen Sie Ihren Linux-Rechner über das Netzwerk administrieren. Daher sind unsere Beispiele in der Regel auf den Zugriff über eine textorientierte Kommandozeile ausgerichtet. Außerdem zeigen wir Ihnen die Konfiguration anhand der Textdateien des Originalprodukts und verzichten weitgehend auf die Möglichkeiten, die die Distributionen an grafischen Tools mitbringen. Halten Sie das bitte nicht für anachronistisch!

Die Nutzung der originalen Konfigurationsdateien bietet Ihnen eine maximale Übertragbarkeit Ihres mühsam gewonnenen Wissens von einer auf die andere Distribution! Außerdem sind auf lange Sicht Konfigurationsdateien mit ausführlichen Kommentaren, die man sich ausdrucken und abheften kann, für die Administration von Servern deutlich besser geeignet als Dialoge, mit denen komplexe Konfigurationen deutlich schwieriger umzusetzen sind. Sind Sie nicht auch schon an Dialogen verzweifelt, die aufgrund der Funktionsvielfalt des zu konfigurierenden Produkts mit Reitern und Schaltern überladen sind und bei denen Sie nach einem halben Jahr wieder vergessen haben, wo das Kästchen mit dem Häkchen war, das eine gerade für Sie wichtige Funktion an und ausschaltet? Vertrauen Sie unserer langjährigen Erfahrung!

Auch lassen sich viele Dinge an Kommandos deutlich leichter erklären als an grafischen Dialogen. In der Regel sind für die Administration über grafische Dialoge Voraussetzungen notwendig, die Ihnen im Notfall nicht zur Verfügung stehen. Der Zugriff über eine Shell stellt ein einheitliches Minimum dar, das Sie beherrschen sollten, um auch in Problemfällen die richtigen und wichtigen Dinge tun zu können. Außerdem sagen Sie Ihrem Hund ja auch nicht mit wildem Armgefuchtel, sondern mit »Sitz!« oder »Komm her!«, dass er sich hinsetzen oder herkommen soll. Und Sie werden sehen, dass es nicht wirklich komplizierter ist, Dinge mit der Kommandozeile zu erledigen. Auch wenn Sie es nicht glauben, auch wir sind tippfaul!

1.5 Argumente gegen ein paar weit verbreitete Missverständnisse

Wir möchten die Gelegenheit nutzen, einigen Missverständnissen entgegenzutreten, mit denen wir bei unserer täglichen Arbeit immer wieder konfrontiert werden – selbst auf die Gefahr hin, dass Sie ohnehin in diesen Fragen schon unserer Meinung sind:

- **Netzwerktechnik ist kompliziert.** Computer unterhalten sich auf eine andere Art und Weise, als wir es tun. Jedoch sind sie von Menschen programmiert worden und die der Computerverständigung zugrunde liegenden Konzepte ba-

sieren auf Konzepten menschlicher Kommunikation. Daher stimmt die Aussage, Netzwerktechnik wäre kompliziert, weil beim Datenaustausch Dinge berücksichtigt werden müssen, die unser Gehirn von alleine leistet. Aber wenn Sie die grundlegenden Konzepte begriffen haben, wird Ihnen das »Netzwerken« auch nicht mehr komplizierter vorkommen als das Autofahren. Im Gegenteil: Sie werden vielen Themen in Funk, Fernsehen und Zeitung besser folgen können, wenn Ihnen die Grundlagen klar sind!

- **Netzwerktechnik ist trockene Materie.** Auch diese Einschätzung teilen wir nicht! Gerade angesichts der Flexibilität, die Linux im Netzwerkbereich bietet, und der vielfältigen Test- und Experimentiermöglichkeiten bereitet es sogar Vergnügen[1], eigene Erfahrungen zu sammeln und Neues auszuprobieren. Wenn unsere Präsentation des Linux-Networking bisweilen unernst wirkt, liegt es daran, dass uns das Thema Spaß macht und wir diesen Spaß mit Ihnen teilen wollen! Und wenn Sie mal das Bedürfnis nach wirklich trockener Materie haben, empfehlen wir Ihnen die Lektüre einschlägiger RFCs[2] oder einen guten trockenen Riesling.
- **Denglisch ist really cool.** Nein, wir finden es absolut uncool und würden am liebsten nur ordentlich eingedeutschte Begriffe verwenden. Aber lesbarer und verständlicher würde unser Buch dadurch nicht, denn die englischen Fachausdrücke haben sich weitgehend durchgesetzt – nicht nur bei den Netzwerkgurus, sondern auch beim durchschnittlichen Administrator und sogar bei vielen normalen Nutzern im deutschen Sprachraum – und wichtige Dokumentation für Linux ist noch immer nur in Englisch verfügbar. Wir wollen also nicht krampfhaft alle englischen Bezeichnungen ins Deutsche übersetzen. Trotzdem versuchen wir, beim Schreiben dieses Buches die Sprachen nicht allzu wild zu mischen.

1.6 Dank und Abbitte

Was lange währt, wird endlich. Ob es gut geworden ist, entscheiden Sie! Wir wollen nicht alle Mitwirkenden an diesem Projekt einzeln aufführen, weil wir den inhaltlichen Teil nicht erst auf Seite 204 beginnen lassen möchten. Aber wir wollen zumindest unserem Lektor Boris Karnikowski für seine Engelsgeduld danken, die er uns »Freizeit-Schreibern« entgegengebracht hat. Ebenfalls eine unverzichtbare Hilfe war Wilhelm Dolle, der mit kritischen Bemerkungen unseren Blick geschärft und auf das Wesentliche gerichtet hat. Und ohne Friederike Daenecke wären zu unseren gedanklichen Sprüngen noch einige grammatische und orthografische hinzugekommen, die Ihnen das Lesen schwer gemacht hätten.

Außerdem danken wir allen Gegenständen, Produkten und Genussmitteln, die die Arbeit an diesem Buch erträglicher gemacht haben. Und dann wären da noch die Mitleidenden aus unserem näheren Umfeld, die den Abschluss dieses Projektes bestimmt dringender erwartet haben als wir selbst. Dass wir uns an dieser Stelle nicht

[1] Echte Linux-Fans sind in puncto Vergnügen ziemlich anspruchslos.

[2] Beispielsweise http://www.ietf.org/rfc/rfc2822.txt

nur für die Unterstützung bedanken, sondern uns auch für die Unbilden entschuldigen wollen, die dieses Projekt während der letzten Jahre über unsere konsensgesicherte Nahwelt gebracht hat, sei ein Index dafür.

Wir hoffen inständig, dass sich unsere Lieben noch nicht allzu sehr daran gewöhnt haben, dass das über der Computertastatur platzsparend in sich zusammengesunkene Wesen kaum ansprechbar ist, und uns demnächst bereitwillig wieder an den häuslichen Aktivitäten teilhaben lassen. Wir geloben Besserung und werden klaglos einkaufen gehen, spülen und bei Hausaufgaben helfen, jedenfalls sobald wir die Festplatte neu organisiert, die geplanten virtuellen Maschinen eingerichtet und einige interessante neue Programme ausprobiert haben.

Aber der Worte sind genug gewechselt: Lassen Sie uns Ihnen Taten zeigen!

2 Mit Linux ins Internet

Wie beginnt man ein Buch über Linux und Netzwerke? Und vor allem, womit? Wir könnten uns natürlich den theoretischen Ansatz zu eigen machen – bedenken Sie, wir arbeiten an der Universität! – und eine mehrbändige Abhandlung zur Netzwerktheorie erstellen. Ausführliche Darstellungen sind einfacher zu schreiben als verständliche. Wir könnten auch den Lauf der (Daten-)Pakete durch Leitungen, Protokollschichten und Tabellen weitschweifig und erschöpfend darstellen. Aber uns beschlich das Gefühl, dass Sie nicht unbedingt viel davon hätten.

Wir haben uns daher für einen eher pragmatischen Ansatz entschieden: Wir wollen Ihnen anhand weniger Kommandozeilen-Tools zeigen, wie man die Netzwerkschnittstellen unter Linux »zu Fuß« konfiguriert. Warum auf der Kommandozeile? Die Tools dafür existieren in jeder Distribution und auch dann, wenn Ihnen Ihre grafische Oberfläche einmal abhanden gekommen ist. So können Sie dann zumindest das Netzwerk konfigurieren und mit einem Textbrowser im Internet nachsehen, warum Ihr X nicht läuft.

Wir werden danach auf die Spezifika einiger ausgewählter Distributionen eingehen. Wenn diejenige, die Sie nutzen, nicht darunter ist, haben Sie aber (hoffentlich) trotzdem etwas von unserem Buch. Außerdem wollen wir Ihnen tiefergehende Erklärungen nicht vorenthalten: Im Anhang ab S. 325 erläutern wir einige technische Grundfunktionen des Netzwerks, sodass Sie bei Bedarf dort nachschlagen können.

2.1 Standardparameter für eine Netzwerkverbindung

Der Linux-Kernel liefert nach der Initialisierung der Netzwerk-Treibermodule eine standardisierte Schnittstelle, über die Anwendungssoftware das Netzwerk nutzen kann. Netzwerk heißt hier »Internet« bzw. IP-Netzwerk (»TCP/IP«). Wenn Sie also Ihren Linux-Rechner mit einem Windows-PC vernetzt haben, werden Sie erst dann über das Netzwerk mit ihm kommunizieren können, wenn beide Rechner TCP/IP verstehen und dafür eingerichtet sind. Linux-Rechner sind ohne eine IP-Adresse nicht in der Lage, ein Windows-Netzwerk zu nutzen. Und auch wenn Sie nur zwei Rechner mit einem Kabel verbinden oder zwischen ihnen eine drahtlose Kommunikation stattfinden soll, muss dafür auf beiden das IP-Netzwerk konfiguriert worden sein.

Was ist dazu notwendig? Gehen wir einmal von einer Standard-Ethernet-Vernetzung aus; von dieser unterscheiden sich die anderen Netzwerkarten in Konfigura-

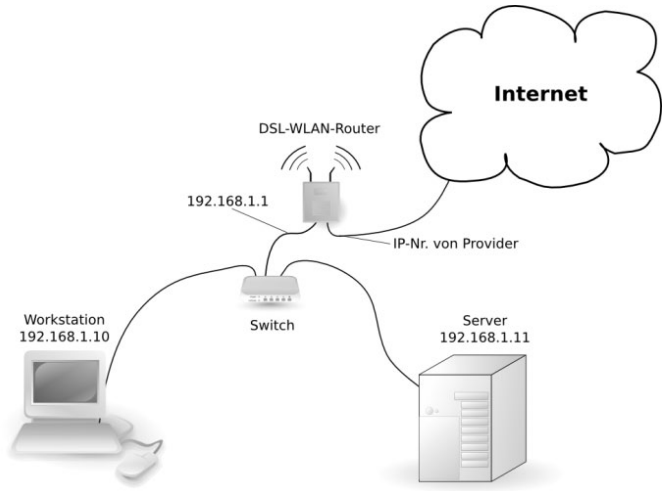


Abbildung 2.1: Subnetz mit zwei Rechnern und einem Router

tion und Nutzung eigentlich nur durch den Namen der Schnittstelle, mit dem sie angesprochen werden.

Damit der eine Rechner weiß, wo er den anderen zu suchen hat, brauchen beide eine **IP-Adresse**. Wie sind IP-Nummern aufgebaut? Bei einer IP-Nummer ist oft der Wert hinter dem letzten Punkt spezifisch für den Rechner; die Werte davor könnte man als Hierarchie eines Netzwerks interpretieren: Der Bereich für private IP-Nummern, den wir in unseren Beispielen benutzen, beginnt mit 192.168. Wir definieren darin Subnetze, die an der dritten Stelle beginnen, z. B. 192.168.1. Ein einzelner Rechner bekommt dann einen für das Subnetz eindeutigen Wert an der vierten Stelle, z. B. 192.168.1.10.

So erfährt das Gegenüber, von wem eine Anfrage kommt und wohin die Antwort gehen muss. Aber man kann nicht grundsätzlich davon ausgehen, dass beide Rechner merken, dass sie im selben Zimmer stehen bzw. an demselben Kabel hängen. Darum muss bei jedem Rechner auch die sogenannte **Netzmaske** gesetzt sein. Mit deren Hilfe kann der Rechner erkennen, wo er das gewünschte Gegenüber zu suchen hat: Liefert eine (logische) Und-Verknüpfung der Netzmaske mit der Zieladresse den gleichen Wert, den das **Netzwerk** hat, so befindet sich der Rechner im gleichen Subnetz. Kommt ein anderer Wert heraus, muss die Anfrage an einen Gateway-Rechner weitergeleitet werden, der sich um die Weiterleitung kümmern soll.

In unserem Beispiel (siehe Abb. 2.1) hat der Rechner, auf dem wir uns befinden, die IP-Adresse 192.168.1.10. Da wir uns im Netzwerk 192.168.1.0 befinden, gelten durch die Definition einer Netzmaske mit dem Wert 255.255.255.0 alle IP-Adressen als benachbart, die mit 192.168.1 beginnen. Für die Netzmaske lässt sich eine einfache Faustregel merken: Ein Wert von 255 bedeutet, dass der an dieser Position in der

eigenen IP-Adresse vorkommende Wert auch bei der Zieladresse gleich sein muss; ein Wert von 0 bedeutet, dass der an dieser Position vorkommende Wert beliebig sein darf. Mit diesen Parametern können wir jetzt die Netzwerkkarte konfigurieren.

Bei einer Modem-Einwahl oder der Konfiguration per DHCP werden diese Daten vom bereitstellenden Host übernommen. Sie lassen sich aber auch mit wenig Aufwand über das Kommando `ifconfig` »zu Fuß« einstellen. Schauen wir dazu erst einmal, was `ifconfig` zur Konfiguration der Netzwerkkarte zu sagen hat:

```
hugo:~# ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0A
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:5
hugo:~#
```

Ihnen wird es noch anverschiedenen Stellen auffallen, dass Kommandos eine gewisse Geschwätzigkeit an den Tag legen. Eine wichtige Kulturtechnik in solchen Situationen ist die Fähigkeit des selektiven Lesens. Wir wollen diese auf unser Beispiel einmal anwenden: Uns interessiert an dieser Stelle nur die Information, dass bei der Hardware-Adresse nicht `00:00:00:00:00:00` steht. Dieser Fall wäre nämlich ein Indiz dafür, dass die Netzwerkkarte nicht korrekt vom Kernel erkannt worden ist und die Hardware-Daten von `ifconfig` nicht ausgelesen werden konnten.

Da dort eine – uns im Moment zwar nicht weiter interessierende – Kombination aus Hexadezimalwerten zu erkennen ist, gehen wir davon aus, dass die Netzwerkkarte ordnungsgemäß vom Kernel erkannt wurde. Die Parameter für den Aufruf von `ifconfig` erklären sich dabei nahezu von selbst: Wir weisen der Netzwerkkarte `eth0` die IP-Adresse `192.168.1.10` zu und geben mit Hilfe der Schlüsselwörter `netmask` und `broadcast` die Netzmaske und die Broadcast-Adresse an. Wenn sich `ifconfig` nicht beschwert, sind die Konfigurationseinstellungen übernommen worden. Zur Kontrolle rufen wir mit `ifconfig eth0` die aktuelle Konfiguration noch einmal ab:

```
hugo:~# ifconfig eth0 192.168.2.10 netmask 255.255.255.0 broadcast 192.168.2.255
hugo:~# ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0A
          inet Adresse:192.168.1.10  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:484 (484.0 b)  TX bytes:0 (0.0 b)
          Interrupt:5
hugo:~#
```

Damit ist die Netzwerkkarte konfiguriert und im Übrigen auch schon aktiviert – was an dem `UP` in der dritten Zeile zu erkennen ist. Hier ist ein »Helferlein« im Kommando `ifconfig` aktiv, der nach der Zuweisung einer IP-Nummer automatisch die Karte »hochfährt«. Sie können die Netzwerkkarte mit `ifconfig eth0 down` deaktivieren, ohne dass sie ihre Konfiguration verliert, und mit `ifconfig eth0 up` mit der alten Konfiguration wieder aktivieren.

Jetzt müssen wir dem System noch mitteilen, wo andere Hosts zu finden sind. Für das **Routing** der Daten, also die Informationen darüber, wo Pakete hingeleitet werden müssen, wenn sie ein bestimmtes Ziel erreichen sollen (siehe auch A.1.3 im Anhang) führt der Kernel eine Tabelle, die wir jetzt eigentlich um einen passenden Eintrag ergänzen müssten. Für das lokale Subnetz erleichtert `ifconfig` uns diese Aufgabe, indem es bereits eine Standardroute einträgt. Ein erster Aufruf des `route`-Kommandos zeigt:

```
hugo:~# route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
hugo:~#
```

Bei der Konfiguration der Netzwerkkarte leitet `ifconfig` den Wert für das Netzwerk von der IP-Adresse ab und trägt diesen mit der angegebenen Netzwerkmaske in die Routing-Tabelle ein. Mit Hilfe dieses Eintrags sollten andere Computer in diesem Netzwerk bereits gefunden werden können. Die Standardmethode, um zu testen, ob ein Rechner im Netzwerk antwortet, ist das »Pingen«:

```
victor@hugo:~$  ping -c 1 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=64 time=28.7 ms

--- 192.168.1.11 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 28.784/28.784/28.784/0.000 ms
victor@hugo:~$
```

Hier haben wir ein Paket (Option `-c 1`) zum Rechner 192.168.1.11 geschickt und ein Paket zurückbekommen. Damit können wir zufrieden sein: Das Netzwerk funktioniert. Sie hatten leider nicht das Glück, einen Output wie im obigen Beispiel zu erhalten? Ab S. 306 geben wir Ihnen einige Tipps für die Fehlersuche.

2.1.1 Der Weg hinaus

Wenn das Netzwerk einen Computer oder DSL-Router besitzt, über den man Anfragen in die Weiten des Internets oder auch einfach nur in andere Subnetze stel-

len möchte, sollte man die Routing-Tabelle um einen Gateway-Eintrag ergänzen. Dann weiß der Client, an welchen Rechner Anfragen weitergegeben werden müssen, wenn auf eine IP-Nummer zugegriffen wird, die nicht zum eigenen Subnetz gehört. Dazu benötigen Sie die IP-Adresse des Rechners, der diese Funktion ausführt, z. B. 192.168.1.1:

```
hugo:~# route add default gw 192.168.1.1
hugo:~# route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
hugo:~#
```

Dieser Eintrag sollte ausreichen, um jetzt auch an Rechner heranzukommen, die jenseits unseres Subnetzes liegen. Wichtig ist natürlich, dass das Gateway diese Informationen auch weiterleitet. Nehmen wir einmal an, hinter der IP-Adresse verbirgt sich unser DSL-Router. Dann sollten wir jetzt einen Ping ins »Internet« machen können:

```
victor@hugo:~$ ping -c 1 www.uni-duesseldorf.de
PING www.uni-duesseldorf.de (134.99.128.40) 56(84) bytes of data.
64 bytes from www.uni-duesseldorf.de (134.99.128.40): icmp_seq=1 ttl=248 time=28.0 ms

--- www.uni-duesseldorf.de ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 28.061/28.061/28.061/0.000 ms
victor@hugo:~$
```

Erreichen Sie den Server nicht, sollten Sie zuerst noch eine andere, Ihnen bekannte Adresse ausprobieren, bevor Sie anfangen, den Fehler bei sich zu suchen. Mitunter haben Internet-Provider ein Problem mit dem Routing, sodass bestimmte Server zeitweise nicht erreichbar sind. Ob der Fehler vor oder hinter und wenn wie weit hinter unserem DSL-Zugang liegt, lässt sich mit dem Tool `traceroute` herausfinden. Damit werden alle Zwischenstationen der Route zum Zielserver ausgegeben; im Fehlerfall kann man dort erkennen, wo es hakt und ob es das noch im eigenen Netz tut:

```
victor@hugo:~$ traceroute www.uni-duesseldorf.de
traceroute to www.uni-duesseldorf.de (134.99.128.40), 30 hops max, 38 byte packets
 1  192.168.1.1 (192.168.1.1)  1.308 ms  0.819 ms  0.797 ms
 2  62.214.64.97 (62.214.64.97)  18.435 ms  18.543 ms  28.768 ms
 3  fra20ip6.versatel.de (213.30.194.230)  22.115 ms  20.794 ms  20.643 ms
 4  FFMGW3.arcor-ip.net (80.81.192.117)  20.063 ms  20.088 ms  20.097 ms
 5  ffm-145-254-16-169.arcor-ip.net (145.254.16.169)  23.011 ms  21.411 ms  20.429 ms
```

```
 6  esn-145-254-12-194.arcor-ip.net (145.254.12.194)  27.672 ms  24.186 ms  25.290 ms
 7  * * *
 8  isgw0003.isis.de (195.158.131.31)  26.399 ms  25.806 ms  25.143 ms
 9  * * *
10  www.uni-duesseldorf.de (134.99.128.40)  27.042 ms  27.736 ms  27.233 ms
victor@hugo:~$
```

Sie sehen in unserem Beispiel, dass der erste Knoten, der angesprochen wird, das in der Routing-Tabelle eingetragene Gateway ist. Danach werden dann die Router angezeigt, die bei den verschiedenen Providern auf dem Weg zur Universität Düsseldorf die Weitergabe der Pakete übernehmen.

Auf dem Weg zum Ziel kann es unter Umständen zu Problemen kommen; entweder liegt der eine oder andere Router dazwischen, der das »Tracen« der Route nicht zulässt, oder eine Route funktioniert nicht. Sie können dies an den Sternen in der Ausgabe von `traceroute` erkennen. Dies ist so lange kein Problem, wie die Pakete einen Weg zum Zielserver und wieder zurück finden. Sollten am Ende nur Sterne ausgegeben werden, liegt dort der Fehler. Besteht z. B. keine Internetverbindung vom DSL-Router zu Ihrem Provider, wird `traceroute` als nächsten Schritt hinter dem Router nur Sterne ausgeben.

Kommen Sie gar nicht weiter – und auch gar nicht an andere Rechner heran – sollten Sie die Routing-Tabelle noch einmal überprüfen: Falsche Routen können Sie mit der `del`-Option des `route`-Kommandos löschen:

```
victor@hugo:~$ route add default gw 192.168.1.2
victor@hugo:~$ ping www.uni-duesseldorf.de
PING www.uni-duesseldorf.de (134.99.128.40) 56(84) bytes of data.
# Nichts passiert, wir unterbrechen mal mit  STRG + C

--- www.uni-duesseldorf.de ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 5000ms
victor@hugo:~$ route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.2     0.0.0.0         UG    0      0        0 eth0
# Ups, das war die falsche IP-Nummer
victor@hugo:~$ route del default gw 192.168.1.2
victor@hugo:~$ route add default gw 192.168.1.1
victor@hugo:~$ route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
victor@hugo:~$
```

Wir haben hier eine falsche IP-Nummer als Gateway angegeben, weswegen unsere Pings nicht aus dem Subnetz hinausgelangen können. Der Blick in die Routing-Tabelle und ein wenig Nachdenken bringen uns die notwendige Erkenntnis; das System kann uns hier nicht helfen, da die angegebene IP-Nummer sich im richtigen Subnetz befindet und etwaige andere Kontrollmechanismen hier nicht zur Verfügung stehen. Das Löschen der falschen Route geschieht mit Hilfe des Schlüsselwortes `del`; die restlichen Optionen sollten Sie der Einfachheit halber genau so angeben, wie Sie es beim Hinzufügen der Route getan haben.

Wenn Sie viele Routen angelegt haben und nicht herausfinden können, welche nun genau ein Problem macht, können Sie die Routing-Tabellen auch dadurch löschen, dass Sie die Netzwerkkarte herunter- und wieder hochfahren.

```
hugo:~# ifconfig eth0 down
hugo:~# route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
hugo:~# ifconfig eth0 up
hugo:~# ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0A
          inet Adresse:192.168.1.10  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:484 (484.0 b)  TX bytes:0 (0.0 b)
          Interrupt:5
hugo:~# route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0      eth0
hugo:~#
```

Ein Blick auf die Netzwerkkonfiguration zeigt, dass der Kernel sich IP-Nummer, die Netzmaske und die Broadcast-Adresse gemerkt hat. Ein einfaches `ifconfig eth0 up` sorgt dafür, dass das Netzwerk wieder in der Grundkonfiguration zur Verfügung steht. IP-Adresse und alle anderen Parameter kann man während des laufenden Betriebs ändern. Alle Änderungen an der IP-Adresse haben aber zur Folge, dass spezielle Routen zu diesem Device in der Tabelle gelöscht werden.

2.1.2 Wenn Namen nicht Schall und Rauch sein sollen

Die Möglichkeit, Rechner über einen Namen anstatt über die IP-Nummer anzusprechen, ist bereits ein weitergehender Service, der für das Funktionieren eines TCP/IP-Netzwerks nicht essenziell ist. Wir sind nur mittlerweile so daran gewöhnt,

dass wir diesen Umstand oft vergessen. Wie können wir uns dieser Bequemlichkeit versichern? Hier gilt es, zwei unterschiedliche Fälle zu beachten:

DNS-Einträge für den Zugriff auf Rechner im Internet: Die Umsetzung von Namen in IP-Adressen ist über den **Resolver** realisiert. Es handelt sich hier im Prinzip um eine Bibliothek, die von allen Programmen benutzt wird, die Zugriffe auf Internet-Hosts durchführen. Der Resolver erwartet einen **Nameserver** oder **DNS-Server**, der auf eine Namensanfrage eine IP-Nummer zurückliefert.

Ein Rechnername funktioniert ähnlich wie eine IP-Nummer, nur dass die Bedeutungshierarchie hier andersherum aufgebaut ist: Es werden sogen. Top-Level-Domains definiert, wie z. B. »de«. Die sogenannten Subdomains werden dann links von der Top-Level-Domain angehängt. wurde an die Universität Düsseldorf die Subdomain »uni-duesseldorf« für den Bereich »de« vergeben, was zusammengesetzt »uni-duesseldorf.de« ergibt. In der Regel müssen für Subdomains eigene DNS-Server eingerichtet werden, die den Top-Level-Domain-Servern bekannt sein müssen. Neue Host-Adressen wie z. B. »www.uni-duesseldorf.de« können dann in Düsseldorf selbst eingerichtet werden. Eine Anfrage aus den Weiten des Internets nach »www.uni-duesseldorf.de« wandert dann über den Top-Level-Domain-Server bis zum DNS-Server der Universität Düsseldorf, der die Anfrage beantworten kann. Der Nameservice besitzt also nicht nur technische, sondern auch organisatorische Aspekte, die bei der Einrichtung beachtet werden müssen.

Für den Anfang genügt es aber zu wissen, dass alle DNS-Server untereinander verbunden sind und wir nur die IP-Adresse des Nameservers unseres Internet-Providers an passender Stelle eintragen müssen, um an den weltweiten Namensraum angebunden zu sein. Die passende Datei dafür ist `/etc/resolv.conf`, die mit wenigen Daten zu füllen ist: Wir benötigen a) eine Information darüber, in welcher Domain wir uns befinden, und b) die IP-Nummern der für uns gültigen DNS-Server:

```
hugo:~# cat /etc/resolv.conf
domain beispiel.org
nameserver 192.168.1.1
nameserver 192.168.1.5
hugo:~#
```

Hinter dem Schlüsselwort `domain` wird der Name der aktuellen Domain eingetragen. Es können auch weitere Domainnamen mit dem Schlüsselwort `search` eingetragen werden. Der Sinn beider Angaben ist eine Vereinfachung der Namensauflösung: Wenn Sie nur einen Hostnamen angeben, wird versucht, die Kombinationen aus Hostname und den in `domain` und `search` angegebenen Domains aufzulösen, wenn zum Hostnamen allein keine IP-Nummer gefunden werden kann.

Wie Sie an unserem Beispiel sehen, kann man mit Hilfe des Schlüsselworts »nameserver« auch mehrere IP-Nummern angeben. Da der Nameservice eine wichtige Funktion darstellt, kann der Ausfall eines DNS-Servers durch die Angabe eines Ausweich-Servers kompensiert werden.

Achtung

Denken Sie daran, an dieser Stelle keinen *Namen* einzutragen! Sonst haben Sie das klassische Henne-Ei-Problem: Ihre Clients müssen erst den Namen auflösen, um den DNS-Server ansprechen zu können. Warum das unter Umständen doch funktioniert – worauf Sie sich aber niemals verlassen sollten – erfahren Sie, wenn wir die Datei `/etc/hosts` besprechen.

Haben Sie die gleichen IP-Nummern eintragen, wie wir sie in unserem Beispiel verwendet haben? Die Wahrscheinlichkeit ist groß, dass das bei Ihnen *nicht* funktioniert. Die Informationen über die für Sie richtigen DNS-Server-Adressen bekommen Sie von Ihrem Netzwerkadministrator oder Ihrem Internetprovider. Vielleicht haben Sie aber auch Glück, oder sich per Modem eingewählt oder die IP-Nummer per DHCP zugewiesen bekommen. In diesem Fall tragen die Programme, die die Verbindung hergestellt und das Netzwerk konfiguriert haben, die Adressen automatisch in die Datei `/etc/resolv.conf` ein.

Um die Funktion Ihrer DNS-Konfiguration zu testen, können Sie mit dem Kommando `nslookup` eine DNS-Anfrage starten. Wenn die Einträge für die DNS-Server richtig sind, erhalten Sie ein Ergebnis wie in unserem Beispiel zusammen mit dem Hinweis, welcher Server Ihnen geantwortet hat.

```
victor@hugo:~$ nslookup www.uni-duesseldorf.de
Server:         192.168.1.1
Address:        192.168.1.1#53

Non-authoritative answer:
Name:   www.uni-duesseldorf.de
Address: 134.99.128.40
victor@hugo:~$
```

Lokale Namen in der Datei /etc/hosts verwalten: Wir hatten eine zweite Möglichkeit erwähnt, die es Ihnen ermöglicht, Namen statt IP-Nummern zu verwenden. Jedes Linux-System – jedes Windows-System übrigens auch – besitzt eine Datei `hosts`, in der in einfacher Form Zuordnungen von IP-Nummern zu Namen gespeichert werden. Die Form der im Verzeichnis `/etc` liegenden Datei ist dabei recht einfach gehalten:

```
victor@hugo:~$ cat /etc/hosts
127.0.0.1       localhost
192.168.1.10    hugo   hugo.beispiel.org
victor@hugo:~$
```

Am Anfang einer Zeile steht eine IP-Nummer, der dann mit einer beliebigen Anzahl von Leerzeichen und Tabulatoren getrennt, eine beliebige Anzahl von Namen

zugeordnet werden können. Diese Namen können lokal auf dem System als Synonyme für die IP-Nummer genutzt werden. Andere Rechner können ohne weitere Hilfsmittel darauf jedoch nicht zugreifen; mit welchem Hilfsmittel das doch funktioniert, ist in Abschnitt 3.5 beschrieben.

Viele Programme erwarten, den Hostnamen, also den Namen, den Sie Ihrem Computer gegeben haben, direkt ansprechen zu können. Daher sollte dieser Name in der Datei `/etc/hosts` eingetragen sein. Sie können ihn, wenn Sie Ihrem Rechner keine eigene IP-Nummer gegeben haben, als Alias für `localhost` eintragen.

Die Datei `/etc/hosts` wird übrigens nicht ausgewertet, wenn Sie eine DNS-Anfrage starten; das obige Beispiel mit `nslookup` funktioniert daher nicht mit Namen, die nur in der `/etc/hosts` eingetragen sind:

```
victor@hugo:~$ nslookup hugo
Server:         192.168.1.1
Address:        192.168.1.1#53

** server can't find hugo: NXDOMAIN
victor@hugo:~$ ping -c 1 hugo
PING hugo (192.168.1.10) 56(84) bytes of data.
64 bytes from hugo (192.168.1.10): icmp_seq=1 ttl=64 time=0.233 ms

--- hugo ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.233/0.233/0.233/0.000 ms
victor@hugo:~$
```

Aber gefunden werden die Rechner trotzdem!

2.1.3 Automatische Konfiguration mit DHCP

Die Netzwerkkonfiguration gestaltet sich wesentlich einfacher, wenn im Netzwerk ein DHCP-Server bereitgestellt wird, der neben einer IP-Nummer auch Informationen über das Standard-Routing und die Nameserver-Konfiguration übermittelt. Wir demonstrieren Ihnen dies einmal mit dem Tool `dhclient`:

```
hugo:~# dhclient eth0
Internet Software Consortium DHCP Client 2.0pl5
Copyright 1995, 1996, 1997, 1998, 1999 The Internet Software Consortium.
All rights reserved.

Please contribute if you find this software useful.
For info, please visit http://www.isc.org/dhcp-contrib.html

Listening on LPF/eth0/fe:fd:00:00:00:0a
Sending on   LPF/eth0/fe:fd:00:00:00:0a
```

```
Sending on   Socket/fallback/fallback-net
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 192.168.1.1
bound to 192.168.1.10 -- renewal in 43200 seconds.
hugo:~# ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0A
          inet Adresse:192.168.1.10  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
[...]
hugo:~# route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
hugo:~# ps ax
<...>
 1117 ?        Ss     0:00 dhclient eth0
 1118 tty1     R+     0:00 ps ax
hugo:~#
```

Das Kommando `dhclient` zeigt Ihnen relativ ausführlich an, was es tut, um an eine IP-Adresse zu gelangen: Nach ein wenig Werbung sehen Sie, auf welcher Schnittstelle mit welcher MAC-Adresse ein `DHCPREQUEST` gestellt wird. Sobald sich ein DHCP-Server gemeldet hat, erhalten Sie mit der Ausgabe von `DHCPACK from 192.168.1.1` darüber Nachricht. Mitunter müssen auch mehr als ein `DHCPREQUEST` abgesetzt werden, um eine Antwort zu erhalten.

Auf eine Besonderheit müssen wir dabei noch hinweisen: Der DHCP-Client beendet sich i. d. R. nicht, sondern läuft im Hintergrund weiter. IP-Nummern werden »geleast«; im Beispiel oben sehen Sie den Hinweis `renewal in xxx seconds`. Damit die IP-Nummer nicht, während der Rechner läuft und sie benutzt, an einen anderen Rechner vergeben wird, sorgt der `dhclient` dafür, dass der DHCP-Server bei Ablauf der Leasing-Zeit darüber informiert wird, dass die IP-Nummer noch belegt ist. Sie sollten den `dhclient` daher nicht ohne Not beenden; die IP-Nummer bleibt Ihnen i. d. R. erhalten, aber Sie haben niemanden mehr, der darauf aufpasst, dass sie Ihnen nicht weggenommen wird. Kollisionen bei doppelt vergebener IP-Nummer sind mitunter sehr schmerzhaft. ;-)

SUSE-Tipp

SUSE hat mit `dhcpcd` eine andere Software installiert, die aber bei der Abfrage genauso funktioniert wie der `dhclient`. `killall dhcpcd` hat aber noch den zusätzlichen Effekt, dass die konfigurierte Netzwerkschnittstelle heruntergefahren wird. Hier ist also kein zusätzliches `ifconfig eth0 down` notwendig. Außerdem verzichtet `dhcpcd` auf die eigentlich unübliche Ausgabe von so viel Text.

2.2 Konfiguration des Netzwerks mit den Tools der Distributionen

Bisher haben wir den mühsamen Fußweg der Netzwerkkonfiguration gezeigt, um Ihnen ein Gefühl für die Mechanismen im Hintergrund zu vermitteln. Die Distributionen vereinfachen die Konfiguration des Netzwerks durch jeweils eigene Tools, die wir jetzt kurz beschreiben wollen.

2.2.1 Debian: Eine einfache Textdatei

Debian verwaltet seine Netzwerkkonfiguration über eine einfache Textdatei namens `interfaces`, die im Verzeichnis `/etc/network` zu finden ist. Diese wird bei der Installation eines Debian-Systems angelegt und mit den Daten gefüllt, die Sie während der Installation angegeben haben.

Debian-Tipp

Sie können das Netzwerk auch dialoggestützt mit `etherconf` konfigurieren. Bei der ersten Installation bzw. bei einem nachträglichen Aufruf von `dpkg-reconfigure etherconf` wird ein Dialog gestartet, in dem Sie die grundlegenden Konfigurationsdaten für die Netzwerkkonfiguration dieses Rechners eingeben können.

Listing 2.1: /etc/network/interfaces

```
auto lo eth0

iface lo inet loopback
iface eth0 inet static
        address 192.168.1.10
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
```

Wenn Sie sich noch an die oben per Hand zu konfigurierenden Parameter erinnern, werden Ihnen die in der Datei hinter den jeweiligen Schlüsselwörtern gespeicherten Werte bekannt vorkommen. Die Konfigurationsdatei ist dabei nach einem einfachen Muster aufgebaut: Das Stichwort `iface` leitet die Definition einer Schnittstelle ein, die hier `eth0` genannt wurde. Auch wenn dieser Name dem Namen der Schnittstelle entspricht, handelt es sich hier um einen logischen Namen; dazu aber später mehr.

Danach werden Typ und Konfigurationsart angegeben. Bei den Typen sollte man sich neben `inet` für normale IPv4-Konfigurationen noch `inet6` für die Konfiguration von IPv6-Schnittstellen merken. Nach den Typen wird die Konfigurationsart angegeben: Hier ist neben `static` noch `dhcp` von Bedeutung; die Option `dhcp` sorgt für eine automatische Konfigurierung der Schnittstelle über DHCP. Bei drahtgebundenem Ethernet müssen dann keine weiteren Konfigurationseinträge gemacht werden.

Zusätzliche Optionen eines `iface`-Eintrags folgen zeilenweise, mit einem Tabulator eingerückt:

- die IP-Adresse (`address`)
- die Netzmaske (`netmask`)
- und die IP-Adresse des Gateways (`gateway`)

Die Broadcast-Adresse ist nicht unbedingt notwendig, rundet die Konfiguration jedoch ab.

Die eigentliche Aufgabe des Hoch- und Herunterfahrens der Netzwerkschnittstellen übernehmen die Scripts `ifup` (für **I**nter**f**ace **up**) und `ifdown` (für **I**nter**f**ace **down**).

```
hugo:~# ifup eth0
hugo:~# ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0A
          inet Adresse:192.168.1.10  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:484 (484.0 b)  TX bytes:0 (0.0 b)
          Interrupt:5
hugo:~# route -n
Kernel IP Routentabelle
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
hugo:~# ifdown eth0
hugo:~#
```

Der Vorgang ist unspektakulär und wird den Fähigkeiten des `ifup`-Kommandos nicht gerecht. Daher wollen wir Ihnen noch ein paar Spezialiäten näherbringen.

Einer Netzwerkkarte mehrere IP-Nummern zuweisen: Ein Linux-Rechner kann mit Hilfe virtueller Netzwerkkarten mehrere IP-Nummern verwalten. Man kann z. B. für den Device `eth0` eine zusätzliche »Netzwerkkarte« mit dem Namen `eth0:1` konfigurieren. Dazu erzeugen wir in der Datei `interfaces` einen zweiten Interface-Eintrag mit den notwendigen Konfigurationsoptionen:

Listing 2.2: /etc/network/interfaces

```
auto lo eth0

iface lo inet loopback
iface eth0 inet static
        address 192.168.1.10
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
```

```
iface zweiter inet static
        address 192.168.1.11
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
```

Bis auf die IP-Nummer und den hinter dem Schlüsselwort `iface` angegebenen Wert `zweiter` unterscheiden sich die Einträge nicht von der Hauptkonfiguration. Wir erwähnten oben, dass der Name hinter dem Schlüsselwort IFACE ein *logischer* Name ist und sich nicht direkt auf eine Schnittstelle bezieht. Das Kommando `ifup eth0` ist dabei eine Kurzform, der intern in `ifup eth0=eth0` umgesetzt wird. Sie können einen anderen Konfigurationsnamen dann z. B. mit `ifup eth0=zweiter` einer realen Schnittstelle zuweisen.

Eine virtuelle Netzwerkschnittstelle `eth0:1` könnte mit unserem neuen Konfigurationseintrag dann auf folgende Weise konfiguriert werden:

```
hugo:~# ifup eth0:1=zweiter
hugo:~# ifconfig eth0:1
eth0:1    Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0A
          inet Adresse:192.168.1.11  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:5
hugo:~#
```

Automatische Konfiguration der Netzwerkkarten beim Systemstart: Es existiert ein Init-Script, das die Netzwerkkarte(n) beim Booten hochfährt. Dieses ruft das Kommando `ifup` mit der Option `-a` auf. Die in diesem Modus hochzufahrenden Schnittstellen sind über die Option `auto` in der Datei `interfaces` gekennzeichnet. Soll die Konfiguration also beim Booten automatisch gestartet werden, muss für diese Kombination noch ein `auto`-Eintrag ergänzt werden:

Listing 2.3: /etc/network/interfaces

```
...
auto eth0:1=zweiter
iface zweiter inet static
        address 192.168.1.11
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
```

Hinter dem Schlüsselwort `auto` können neben der im Beispiel gezeigten vollständigen Darstellung auch Faulenzer-Einträge genutzt werden. Dann bedeutet `auto eth0`, dass die Schnittstelle `eth0` mit der Konfiguration `eth0` hochgefahren wird.

Optionen für WLAN-Karten angeben: Wir werden auf die Konfiguration von WLAN-Karten weiter unten noch ausführlicher eingehen. An dieser Stelle sei aber

vielleicht noch ein Hinweis erlaubt: Die Konfiguration von WLAN-Karten benötigt über die Standardparameter hinaus zusätzliche Angaben, die für die Erkennung des Access-Points und eventuell für eine verschlüsselte Übertragung notwendig sind. Um diese nutzen zu können, muss das Paket `wireless-tools` installiert sein.

Nach der Installation kann die Konfiguration um weitere Parameter ergänzt werden: Für ein WLAN-Netzwerk mit der ESSID (**E**xtended **S**ervice **S**et **Id**entifier) `WLAN`, könnte man eine DHCP-Konfiguration einrichten, die folgende zusätzliche Parameter enthält:

Listing 2.4: /etc/network/interfaces

```
...
iface wlan0 inet dhcp
        wireless-essid WLAN
        wireless-mode Ad-Hoc
```

Handelt es sich bei der WLAN-Hardware z. B. um eine Cardbus-WLAN-Karte, die als Device `wlan0` erkannt wird, wird nach dem Einschieben der Karte die Konfiguration `wlan0` automatisch aktiviert. Der Konfigurationstyp `dhcp` sorgt dafür, dass nach der Konfiguration der WLAN-Karte ein DHCP-Request abgesetzt wird, um die IP-Nummer und weitere Konfigurationsdaten zu erhalten.

2.2.2 SUSE: mit YaST alles unter Kontrolle

Bei SUSE haben Sie neben der Bearbeitung von Textdateien die Möglichkeit, die Netzwerkeinstellungen über YaST vorzunehmen. Die Basiskonfiguration einer Netzwerkkarte wird über das Menü NETZWERKGERÄTE/NETZWERKKARTE aufgerufen. Dort können Sie entweder neue Netzwerkkarten konfigurieren oder über den Button ÄNDERN eine Liste der bereits konfigurierten Geräte aufrufen.

Wenn Sie eine der dort genannten Netzwerkkarten auswählen und über den Button BEARBEITEN den Konfigurationsdialog aufrufen, können Sie dort die Parameter einstellen oder ändern, die wir am Anfang des Kapitels besprochen haben.

Einstellungsmöglichkeiten wie die Angabe des Gateways oder weitere spezielle Routing-Angaben können Sie über den Button ROUTING festlegen. Den Hostnamen, die Domain und die DNS-Server tragen Sie wie zu erwarten mit Hilfe des Menüs RECHNERNAME UND NAMESERVER ein.

Widmen Sie Ihre Aufmerksamkeit noch dem Menü ERWEITERT: Dort können Sie über den Unterpunkt HARDWAREDETAILS feststellen, welches Kernelmodul geladen wird, um die Netzwerkkarte dem System verfügbar zu machen.

In einigen Fällen kann ein und dieselbe Karte über verschiedene Kernelmodule angesteuert werden. Wenn es mit dem automatisch erkannten und der Netzwerkkarte zugewiesenen Modul Probleme gibt, können Sie hier ein anderes auswählen. Außerdem finden Sie hier ein Eingabefeld, mit dem Sie diesem Modul Optionen für den Ladevorgang mitgeben können, falls dies für das fehlerfreie Funktionieren der Karte notwendig ist. Ein Beispiel dafür wären alte ISA-Netzwerkkarten, denen noch eine Angabe für den Interrupt und die Speicheradresse mitgegeben werden müssen.

Unter ERWEITERT/BESONDERE EINSTELLUNGEN haben Sie unter anderem die Möglichkeit festzulegen, dass die Netzwerkkarte beim Start automatisch hochgefahren und konfiguriert werden soll. Diese Option wird bei der Hardware-Erkennung automatisch aktiviert. Wenn Ihnen aber der Sinn danach steht, dieses manuell im laufenden Betrieb durchzuführen – was unter Umständen bei Zweit- und Dritt-Netzwerkkarten sinnvoll sein kann –, können Sie das automatische Hochfahren an dieser Stelle abstellen. Zum Schluss sei noch auf den Dialog ERWEITERT/ZUSÄTZLICHE ADRESSEN hingewiesen: Er erlaubt Ihnen die Vergabe von mehreren IP-Nummern für ein Netzwerkdevice.

Das manuelle Hochfahren einer Netzwerkkarte führen Sie mit dem Kommando `ifup` durch; `ifdown` entsprechend für das Herunterfahren.

```
hugo:~# cat /etc/sysconfig/network/ifcfg-eth0
BOOTPROTO='static'
BROADCAST='192.168.1.255'
IPADDR='192.168.1.10'
MTU=''
NETMASK='255.255.255.0'
NETWORK='192.168.1.0'
REMOTE_IPADDR=''
STARTMODE='auto'
UNIQUE='bSAa.IQxIdIhhuH7'
USERCONTROL='no'
_nm_name='bus-pci-0000:00:0a.0'
hugo:~# ifup eth0
    eth0
hugo:~# ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse FE:FD:00:00:00:0C
          inet Adresse:192.168.1.10  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 Sendewarteschlangenlänge:1000
          RX bytes:2824 (2.7 Kb)  TX bytes:0 (0.0 b)
          Interrupt:5
hugo:~# ifdown eth0
```

```
hugo:~# ifconfig
lo        Protokoll:Lokale Schleife
          inet Adresse:127.0.0.1  Maske:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 Sendewarteschlangenlänge:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
hugo:~#
```

Unser Kommandozeilen-Beispiel enthüllt mit dem ersten Befehl, wo die jeweiligen Konfigurationsdateien gespeichert sind (nämlich in `/etc/sysconfig/network`) und wie die Dateinamen aufgebaut sein müssen (die Konfigurationsdateien beginnen mit `ifcfg-`). Geben Sie nach dem Namen der Schnittstelle keinen Konfigurationsnamen an, wird versucht, eine gleichnamige Konfigurationsdatei dafür zu verwenden. Das Kommando `ifup eth0` konfiguriert dann die Schnittstelle `eth0` folgerichtig mit der Konfiguration aus der Datei `ifcfg-eth0`.

Es können aber auch Konfigurationen angelegt werden, die nicht den Namen der Schnittstelle tragen. Denkbar wäre auf einem Laptop eine Konfiguration `ifcfg-home` und `ifcfg-buero`, die dann jeweils mit `ifup home eth0` respektive `ifup buero eth0` aufgerufen werden kann. Beim Herunterfahren ist jeweils nur die Angabe der Schnittstelle notwendig; die Konfiguration, mit der die Netzwerkkarte hochgefahren wurde, wird automatisch gespeichert und beim Herunterfahren berücksichtigt.

Die Schlüsselwörter, mit denen Sie eigene `ifcfg-*` Dateien erzeugen können, finden Sie über die Hilfeseite `man ifcfg`. Am einfachsten ist es aber auch hier, wenn Sie aus bestehenden Konfigurationen abschreiben. :-)

2.2.3 Systemkonfiguration für Fedora-basierte Systeme

In Fedora-basierten Systemen existieren verschiedene `system-config-`-Scripts auf Python-Basis, die Ihnen ein grafisches Userinterface für die Konfiguration bieten. Für das Netzwerk ist das Script `system-config-network` zuständig.

Hier können Sie über die verschiedenen Reiter die einzelnen Elemente der Netzwerkkonfiguration bearbeiten (Abb. 2.2). Da dieses Script die Konfigurationsdateien des `/etc`-Verzeichnisses verändert, benötigen Sie root-Rechte, um es auszuführen. Die Übersicht, die beim Start angezeigt wird (GERÄTE), enthält die erkannten Netzwerkkarten. Ein Doppelklick auf das Gerät in der Liste öffnet dann einen Dialog, in dem Sie die Konfiguration bearbeiten können (Abb. 2.3).

Die Eingabefelder sind i. d. R. selbsterklärend; das Ausfüllen sollte Ihnen keine Schwierigkeiten bereiten. Ein Hinweis vielleicht noch: Der Reiter ROUTE in diesem Dialog ist für Sie nur interessant, wenn Sie nicht mit den Standardwerten arbeiten, die beim Hochfahren der Karte automatisch gesetzt werden. In der Regel werden Sie dort aber keine Ergänzungen machen müssen, insbesondere wenn Ihr Rechner per DHCP konfiguriert wird.

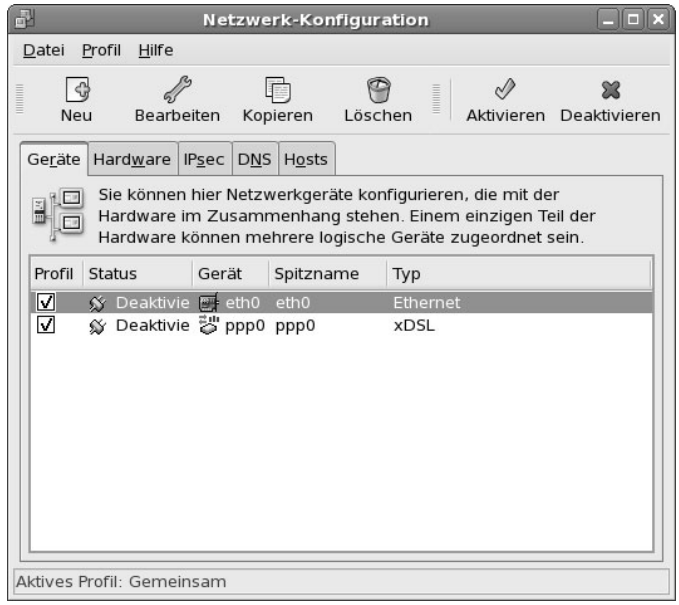


Abbildung 2.2: Fedora: Netzwerkkonfiguration

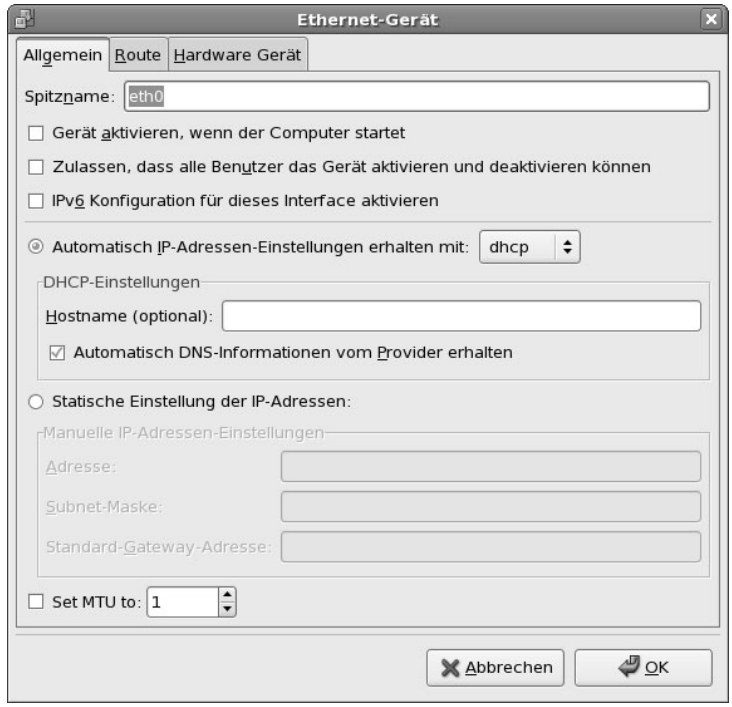


Abbildung 2.3: Fedora: Konfigurationsdialog für eine Netzwerkkarte

2.2.4 Möglichst einfach: der NetworkManager

Ein Unterprojekt des GNOME-Projekts ist der NetworkManager, der passend zur Philosophie des Mutterprojekts ein möglichst einfaches und in der Komplexität stark reduziertes User-Interface für die Netzwerkkonfiguration schaffen soll. Dieser ist in den verschiedenen Fedora-basierten Systemen integriert, wird aber in allen Distributionen, auch in SUSE oder Debian, angeboten.

Der NetworkManager besteht aus drei Server-Komponenten, die mit dem dbus-System arbeiten und mit einem Applet kommunizieren, das in der jeweiligen grafischen Oberfläche läuft. Die Server-Komponenten sorgen dabei für die Erkennung der Geräte, während Sie mit Hilfe des jeweiligen Applets die gefundenen Netzwerke konfigurieren können. Hier werden so viele Automatismen wie möglich angewendet, um Ihnen lästige Konfiguration abzunehmen.

Mit einem Klick auf das Applet erhalten Sie einen Überblick über die gefundenen Netzwerke; dies ist insbesondere bei Funknetzwerken nützlich. Solange Ihre Netzwerkkarte nicht verkabelt ist und einen sogenannten Link hat – d. h., dass auf der anderen Seite des Kabels Hardware angeschlossen ist –, ist der Eintrag im Dropdown-Menü für das KABELNETZWERK grau (Abb. 2.4).

In dem Moment, in dem Sie das Netzwerkkabel anschließen, aktiviert der NetworkManager die Default-Konfiguration, die Sie mit `system-config-network` definiert haben. Ändern Sie dort eine Einstellung, müssen Sie dies dem NetworkManager durch einen Restart mit `/etc/init.d/NetworkManager restart` mitteilen, da er die Konfigurationen für das kabelgebundene Netzwerk nur beim Starten einliest.

In unserer Beispielgrafik sehen Sie in der Taskleiste neben den »600 MHz« zwei Applets, die den Status der Netzwerkverbindung anzeigen. Die Parameter, mit denen das Netzwerk konfiguriert ist, werden Ihnen in einem kleinen Dialog angezeigt, wenn Sie auf das jeweilige Symbol klicken. Nicht aktive Netzwerkschnittstellen

Abbildung 2.4: NetworkManager

werden mit einem kleinen roten Kreuz gekennzeichnet (das rechte von beiden). Das linke Symbol gibt den Status eines Funknetzwerks an; dazu wird neben den beiden Computersymbolen ein Symbol für die Qualität der Funkverbindung angezeigt, das Sie vielleicht von Ihrem Handy kennen.

Einen anderen Weg geht der NetworkManager bei der Konfiguration eines WLANs: Sie sehen in der Übersicht alle *sichtbaren* Funknetzwerke; sobald ein WLAN-Router aber so eingerichtet ist, das die Netzwerk-ID, die sogenannte **ESSID** (**E**xtended **S**ervice **S**et **Id**entifier), nicht öffentlich sichtbar ist oder mit verschlüsselter Übertragung arbeitet, konfigurieren Sie das Netzwerk über den Menüpunkt NEUES FUNKNETZWERK ERSTELLEN. Mit dem Menüpunkt MIT EINEM ANDEREN FUNKNETZWERK VERBINDEN können Sie einem der erkannten Netzwerke zusätzliche Konfigurationsoptionen mitgeben. Dort haben Sie dann z. B. die Möglichkeit, die Keys für die Verschlüsselung anzugeben. Zum Thema WLAN wollen wir Ihnen im nächsten Kapitel aber noch etwas ausführlichere Informationen geben.

2.3 Wenn's funkt: Einrichtung einer WLAN-Verbindung

Kabel in der Wohnung zu verlegen ist ja nicht jedermanns Sache und führt oft zu Störungen – zumindest in der Beziehung. In der Beziehung sind WLAN bzw. Wireless LAN Netzwerke natürlich beziehungsfreundlicher, da sie optisch weniger auffällig sind und nicht beim Staubsaugen stören. Und da mittlerweile jeder bessere Laptop eine WLAN-Karte integriert hat, ist der Aufbau eines Funknetzes auch keine teure Angelegenheit mehr.

WLAN-Karten unterscheiden sich von normalen Netzwerkkarten durch einen – etwas unscharf gesagt – zusätzlichen Funkadapter, der die Verbindung zu einem Access-Point, also einer passenden Gegenstation, herstellt und dann über die Funkverbindung eine Art Ethernet simuliert. Bei der Konfiguration unter Linux schlägt sich das darin nieder, dass für die Funkverbindung ein neues Kommando `iwconfig` existiert, mit dem der Karte die für das Funknetzwerk spezifischen Einstellungen mitgeteilt werden. Ist die Verbindung hergestellt, kann die Karte dann auf die übliche Art und Weise konfiguriert werden; entweder über DHCP oder mit fester IP-Nummer und allem, was dazu gehört.

2.3.1 Kernelmodule für WLAN-Karten

Aber vor die Konfiguration hat der Hersteller die Erkennung der Karte durch den Kernel gesetzt; und da herrscht bei vielen Anbietern noch großes Chaos: Zwei CardBus-Karten eines wirklich namhaften Netzwerkkarten-Herstellers, deren Produkt-IDs sich nur durch einen Buchstaben am Ende unterschieden, enthielten komplett unterschiedliche Chipsätze. Der Boom im WLAN-Bereich treibt diese Blüten und die Kommunikationsbereitschaft der Hersteller gegenüber den Kernel-Entwicklern ist für einen nicht unbedeutenden Teil der verwendeten Chipsätze leider eher gering. Daher sollten Sie es sich zur Pflicht machen, vor dem Kauf einer WLAN-Karte zu prüfen, ob diese von Ihrer Distribution unterstützt wird. Eine Hilfe bietet das

Wireless LAN Howto von Jean Tourrilhes[1], das auch eine ausführliche Übersicht über die zurzeit unterstützten WLAN-Karten bietet.

Es hat jedoch den Anschein, als würde sich die Situation in naher Zukunft verbessern: Die Firma Devicescape hat ihren »Advanced Datapath Driver« unter eine Open-Source-Lizenz gestellt und strebt eine Integration in die Kernel der 2.6er-Serie an. Außerdem wurde der »Wireless Stack« auf dem 2006er Kernel Summit intensiv diskutiert. Der Bereich der WLAN-Unterstützung hat bei den Entwicklern im Moment eine hohe Priorität.

Haben Sie bereits WLAN-Hardware und müssen diese ans Laufen bringen, dann ist bei den freien Distributionen i. d. R. das Kompilieren des passenden Kerneltreibers angesagt. Aufgrund lizenzrechtlicher Probleme bieten Fedora, SUSE und Debian für bestimmte Kernelmodule keine vorgefertigten Pakete an. Wesentlich besser sieht die Unterstützung bei den käuflich zu erwerbenden Linux-Varianten aus: Hier werden auch Treiber beigelegt, die bezüglich der Offenlegung des Quellcodes nicht dem strengen Open-Source-Standard entsprechen.

Wenn Sie leider kein fertiges Paket für den von Ihrer Distribution verwendeten Kernel bekommen können, müssen Sie das Modul selbst kompilieren. Das hört sich schwieriger an, als es ist. Der Kernel enthält ein System, das die Erzeugung von Modulen stark vereinfacht. Außerdem bieten einige Distributionen Tools, um das Erzeugen von Kernelmodulen zu erleichtern. Die Debian-basierten Distributionen bieten mit dem Module-Assistant ein solches Tool, das die Erzeugung des Kernelmoduls aus für Debian vorbereiteten Quellpaketen deutlich erleichtert.

Was benötigen Sie an bereits installierter Software, um ein Kernelmodul selbst kompilieren zu können? Das Wichtigste ist i. d. R. der passende Compiler zum laufenden Kernel. Die Ausgabe von `/proc/version` hilft Ihnen dabei, die richtige Version herauszufinden, wenn Ihre Distribution Ihnen mehrere Compiler zum Installieren zur Verfügung stellt:

```
victor@hugo:~$ cat /proc/version
Linux version 2.6.17-2-vserver-686 (Debian 2.6.17-9) (waldi@debian.org)
(gcc version 4.1.2 20060901 (prerelease) (Debian 4.1.1-13))
#1 SMP Wed Sep 13 18:41:34 UTC 2006
victor@hugo:~$
```

Hier sehen Sie einen Kernel des Debian-Releases Etch, der mit dem Compiler `gcc` in der Version 4.1.2 erzeugt worden ist. Damit Modul und Kernel richtig zusammenpassen, müssten wir in diesem Fall ebenfalls den gcc in der Version 4.1 installieren.

[1] Den zugehörigen URL finden Sie am Ende dieses Abschnitts.

Software und Dateien, die für das Kompilieren von Kernelmodulen notwendig sind	Debian	Fedora	SUSE
Kernel-Header Die zum laufenden Kernel gehörenden Teile des Source-Trees, die für das Kompilieren von Modulen benötigt werden	linux-headers-n.nn.nn-*-*	kernel-devel-n.nn.nn-*-*	kernel-source
C-Compiler Der zum Kernel passende Compiler	gcc-n.n	gcc	gcc
KBuild-Umgebung Die Build-Umgebung, die zum Erzeugen der Module genutzt wird	linux-kbuild-n.nn.nn	in kernel-devel enthalten	in kernel-source enthalten

Tabelle 2.1: Paketübersicht

Das Bauen eines Pakets geht dann i. d. R. fast von selbst: Die Hauptquelle für die notwendigen Kommandos sind Dateien mit den Namen `README` oder `INSTALL`. Vor dem Kompilieren muss der Ort für die Linux-Header und die Konfigurationsdatei des Kernels bekannt sein; diese Informationen sind aber ab Kernel-Version 2.6 mit einem Link mit Namen `build` im Modulverzeichnis des Kernels präsent. Dies wird von den mit dem Modul mitgelieferten Scripts i. d. R. automatisch erkannt, so dass die Kompilierung des Kernelmoduls anhand des Dreisprungs `make config`, `make` und `make install` gelingen sollte.

Nach Ausführung dieser drei Kommandos ist das frisch kompilierte Kernelmodul an die passende Stelle kopiert und sollte mit Hilfe von `modprobe <Modulname>` geladen werden können. Die letzten Zeilen der Ausgabe, die das Kommando `dmesg` erzeugt, sollten Ihnen dann einen Hinweis darauf geben, ob das gerade geladene Kernelmodul auch seine Arbeit verrichtet. Außerdem gibt Ihnen `iwconfig` im Erfolgsfall die neue Schnittstelle aus, die jetzt auf ihre Konfiguration wartet.

Weitere Literatur

- Jean Tourrilhes' »Wireless LAN Ressources for Linux« ist die umfassendste Dokumentation zum Thema Linux und Wireless LAN: `http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/`

2.3.2 Windows-Treiber unter Linux nutzen

Falls Ihre Hardware unter Linux nicht unterstützt werden sollte, gibt es eine Variante, mit der Sie die original Windows-Treiber mit Ihrem Kernel zum Laufen bringen. Wir sind dieser Methode jedoch nicht wirklich zugeneigt und halten sie nur für die Ultima Ratio, falls gar nichts anderes funktioniert.

Windows-Treiber unter Linux mit dem NDIS-Wrapper nutzen	Debian	Fedora	SUSE
NDIS-Wrapper Die kernelspezifischen Teile des NDIS-Wrappers	ndiswrapper-source (muss kompiliert werden)	—	ndiswrapper
Kommandozeilen-Tools die für die Nutzung gebraucht werden	ndiswrapper-utils	—	im Paket ndis-wrapper enthalten
pciutils werden vom NDIS-Wrapper genutzt, um Geräte zu erkennen	pciutils	pciutils	pciutils

Tabelle 2.2: Paketübersicht

Insbesondere die freien Varianten der Distributionen haben ein Problem mit dem Einsatz des NDIS-Wrappers. Daher muss das Modul z. B. unter Debian erst kompiliert werden – was aber mit dem Module-Assistant problemlos erledigt werden kann. Fedora-Nutzer müssen den NDIS-Wrapper aufgrund lizenzrechtlicher Bedenken der Distributoren ganz selbst kompilieren (einen Link zu einer Anleitung haben wir unten aufgeführt).

Der NDIS-Wrapper emuliert die Windows-API im Kernel und ermöglicht es, einen für Windows erstellten Gerätetreiber zu nutzen. Haben Sie die Treiberdateien ausgepackt in einem Verzeichnis liegen, kann das Kommando `ndiswrapper` die mitgelieferte `<treibername>.inf`-Datei zur Installation des Treibers in Ihrem Linux-System nutzen. Eine genaue Anleitung finden Sie in dem unten angegebenen Link. Erlauben Sie uns aber, darauf hinzuweisen, dass Sie mit der Nutzung des NDIS-Wrappers die Stabilität Ihres Linux-Kernels von der Stabilität eines nicht für das System geschriebenen Treibers abhängig machen. Ein direkt von Linux unterstütztes Gerät wird hier deutlich besser und stabiler arbeiten, auch wenn manch einer Ihnen etwas anderes erzählen möchte.

Weitere Literatur

- Allgemeine Installations- und Nutzungsanleitung des NDIS-Wrapper-Projekts: `http://ndiswrapper.sourceforge.net/mediawiki/index.php/Installation`
- Installationsanleitung für Fedora-Nutzer: `http://ndiswrapper.sourceforge.net/mediawiki/index.php/Fedora`

2.3.3 Zur Technik des Funkens

Für die drahtlose Vernetzung existiert mit dem IEEE 802.11 ein Standard, der diesen Namen eigentlich nicht verdient. So gibt es verschiedene »Erweiterungen« dieses Standards, die jeweils unterschiedliche Arten des Datenaustauschs definieren. Die verbreitetsten sind 802.11b für eine Übertragung mit 11 MBit/s und 802.11g für eine Datenrate von 54 MBit/s. Die Versprechungen bezüglich höherer Datenraten vonseiten der Hersteller beziehen sich i. d. R. auf herstellerspezifische Erweiterungen,

die nicht dem Standard entsprechen und daher zu den Produkten anderer Hersteller nicht kompatibel sind. Eine Erweiterung des Standards mit dem Namen 802.11n und einer Datenrate von 540 MBit/s ist in Planung und soll 2008 fertig sein. Wann dann Produkte auf den Markt kommen, die diesen Standard unterstützen, bleibt abzuwarten.

Funknetze sind per se eine unsichere Sache, da der Datenaustausch über ungerichtete Funkwellen verläuft und im Prinzip von jedem abgehört werden kann. Sicherheit bringt hier nur eine Verschlüsselung des Datenverkehrs. Die ursprünglich im Standard vorgesehene Methode WEP hat sich bereits nach kurzer Zeit als unsicher herausgestellt. Kann der Angreifer eine ausreichende Menge an Datenpaketen abhören – je nach Methode zwischen 500.000 und 15 Millionen, kann der Schlüssel auf einem aktuellen PC in weniger als einer Minute geknackt werden.

Achtung

Letzte Urteile über offene Funknetze ordnen deren Betreiber als Provider mit den dazugehörigen Pflichten und Verantwortlichkeiten ein. Das führt zu einer Mithaftung im Falle von Missbräuchen, die mit Hilfe des Internetzugangs über das jeweilige Funknetz ausgeübt wurden. Wir können daher nur vom Betrieb eines offenen Funknetzes abraten!

Der neue, sichere Standard ist WPA bzw. WPA2. Dieser wird jedoch noch nicht von jeder Hardware unterstützt. Für Linux gibt es das Tool `wpa_supplicant`, mit dem beide Standards genutzt werden können. Steht Ihnen für Ihre Hardware der WPA-Standard nicht zur Verfügung, können Sie sich die Verschlüsselung über WEP sparen und sollten Ihre Energie lieber in die Einrichtung eines VPNs investieren, mit dem Sie dann nicht nur den Datenverkehr auf softwarebasis verschlüsseln, sondern auch den Zugang zum Internet kontrollieren können.

Weitere Literatur

- Webseiten zum »Linux WPA/WPA2/IEEE 802.1X Supplicant« inklusive einer Liste der zurzeit unterstützten Hardware: `http://hostap.epitest.fi/wpa_supplicant/`

2.3.4 Besonderheiten bei der Konfiguration

Wir wollen an dieser Stelle nur auf die Anbindung an ein bestehendes Funknetzwerk eingehen. Bei der Konfiguration eines Linux-Systems als Access-Point benötigen Sie noch Wissen, das wir erst im Laufe des Buches vermitteln werden.

Was ist wichtig, um eine Verbindung aufzubauen?

- **Der Name der Schnittstelle**. Sie haben richtig gelesen: Die Schnittstellennamen unterscheiden sich leider je nach Lust und Laune des Treiberprogrammierers.

Beispiele für Namen sind `wlan0` oder bei Karten mit dem Atheros-Chip `ath0`. Bei manchen Treibern wird die Karte auch einfach über einen `eth`-Namen angesprochen. Am einfachsten finden Sie dies durch einen Aufruf des Kommandos `iwconfig` heraus, das Ihnen alle WLAN-fähigen Geräte anzeigt.

- Sie benötigen den Namen des Netzwerks, die sogen. **ESSID**. Bei offenen Netzwerken ist diese sichtbar – d. h. der Name wird Ihnen angezeigt, wenn Sie sich in Reichweite des Access-Points befinden.
- Dann muss Ihr WLAN-Adapter im richtigen Modus arbeiten. Für eine passive Nutzung des Netzwerks sind nur zwei Modi interessant: **Managed** und **Ad-Hoc**. Sie müssen sich aber darüber i. d. R. keine Gedanken machen, da Ihr Adapter diese Modi automatisch erkennen kann.
- Wenn der Austausch mit dem Access-Point verschlüsselt erfolgt, benötigen Sie noch die notwendigen **WEP**- oder **WAP**-Informationen. Dies können Passwörter oder in Hexadezimal- oder ASCII-Notation angegeben Schlüssel sein.

Die Konfiguration einer WLAN-Karte nehmen Sie mit dem Tool `iwconfig` vor. Wir werden Ihnen das an einem Beispiel einer als `wlan0` erkannten WLAN-Karte und dem per WEP verschlüsselten Funk-Netzwerk »LAGU« demonstrieren:

```
hugo:~# iwconfig wlan0
wlan0     IEEE 802.11g  ESSID:""
          Mode:Managed  Channel:0  Access Point: Not-Associated
          Bit Rate:0 kb/s   Tx-Power:14 dBm   Sensitivity=0/3
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality=0/94  Signal level=-95 dBm  Noise level=-95 dBm
          Rx invalid nwid:681  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
hugo:~# iwconfig wlan0 essid LAGU key 0101-2323-3434-5656-6767-8989-00
hugo:~# iwconfig wlan0
wlan0     IEEE 802.11g  ESSID:"LAGU"
          Mode:Managed  Channel:0  Access Point: Not-Associated
          Bit Rate:0 kb/s   Tx-Power:14 dBm   Sensitivity=0/3
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:0101-2323-3434-5656-6767-8989-00   Security mode:restricted
          Power Management:off
          Link Quality=0/94  Signal level=-95 dBm  Noise level=-95 dBm
          Rx invalid nwid:681  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
hugo:~#
```

Mit dem ersten Kommando schauen wir uns an, welche Situation wir im Moment vorfinden. Der 2. Aufruf setzt für den Device `wlan0` die `essid LAGU` und einen in Hexadezimal-Notation angegebenen Schlüssel (`key`) auf den Wert `0101-2323-3434-5656-6767-8989-00`. Damit ist unsere WLAN-Karte ausreichend konfiguriert. Den `Mode` könnte

man an dieser Stelle noch auf `auto` setzen, aber die Voreinstellung `Managed` ist für unser – und die meisten anderen Funknetzwerke passend. In der Regel fängt die Karte jetzt von selbst an, sich einen Gegenpart zu suchen. Tut sie das einmal nicht, müssen Sie sie vielleicht noch »hochfahren«. Das tun Sie mit dem Kommando `ifconfig` so, wie wir es oben bei den Ethernet-Karten gezeigt haben:

```
victor@hugo:~$ ifconfig wlan0 up
victor@hugo:~$ iwconfig wlan0
wlan0     IEEE 802.11g  ESSID:"LAGU"
          Mode:Managed  Frequency:2.432 GHz  Access Point: 00:0F:12:34:56:78
          Bit Rate:48 Mb/s   Tx-Power:17 dBm   Sensitivity=0/3
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:0101-2323-3434-5656-6767-8989-00   Security mode:restricted
          Power Management:off
          Link Quality=36/94  Signal level=-59 dBm  Noise level=-95 dBm
          Rx invalid nwid:681  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
victor@hugo:~$ dhclient wlan0
Internet Systems Consortium DHCP Client V3.0.4
Copyright 2004-2006 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

wifi0: unknown hardware address type 801
wifi0: unknown hardware address type 801
Listening on LPF/wlan0/00:89:78:67:56:45
Sending on   LPF/wlan0/00:89:78:67:56:45
Sending on   Socket/fallback
DHCPREQUEST on ath0 to 255.255.255.255 port 67
DHCPACK from 192.168.1.1
bound to 192.168.1.101 -- renewal in 33152 seconds.
victor@hugo:~$
```

Wenn Sie keine Frequenz angegeben haben, wird die Karte jetzt alle im Standard definierten Kanäle daraufhin durchprobieren, ob irgendwo ein Access-Point antwortet. Wenn Sie – eventuell nach etwas Warten und mehreren beobachtenden `iwconfig`-Aufrufen – hinter dem Schlüsselwort `Access Point` eine MAC-Adresse entdecken können, hat Ihre WLAN-Karte die Verbindung hergestellt. Jetzt müssen Sie sich nur noch eine IP-Nummer besorgen, was wir in unserem Beispiel per DHCP mit Hilfe des `dhclient` getan haben.

Wir haben Ihnen hier den Umgang mit der eher unsicheren WEP-Verschlüsselung gezeigt, um Ihnen zu demonstrieren, dass sich hinter der Konfiguration des Funknetzwerks genauso wenige Geheimnisse verbergen wie bei der normalen Netzwerkkonfiguration. Die Distributionen bieten Ihnen natürlich komfortablere Mittel, um mit wenig Aufwand ein Funknetzwerk ans Laufen zu bringen:

- Für alle Netzwerktypen besitzen die Debian-Tools `ifupdown` eine Erweiterung, die es Ihnen erlaubt in der Datei `/etc/network/interfaces` die zusätzlich notwendigen Konfigurationseinträge abzulegen. Nähere Informationen dazu sind im Debian-Wiki (`wiki.debian.org`) unter den Stichwörtern **WiFi** und **WPA** oder in den `/usr/share/doc`-Verzeichnissen der installierten Pakete zu finden.
- SUSE-Nutzer können die Konfigurationseinträge entweder in YaST vornehmen oder
- wie die Fedora-Nutzer und natürlich alle anderen, die den NetworkManager installiert haben, dessen Funktionen nutzen. Wir haben oben schon erwähnt, dass alle offenen Funknetzwerke von den jeweiligen NetworkManager-Applets bereits im Kontextmenü angezeigt werden. Nicht angezeigte Netzwerke können Sie über VERBINDUNG ZU ANDEREM DRAHTLOSEN NETZWERK HERSTELLEN... konfigurieren und aktivieren.

2.4 Einwahl mit PPP

Für die autorisierte Netzwerkanbindung hat sich mittlerweile PPP, das **P**oint **t**o **P**oint **P**rotocoll, zu einem Standard entwickelt. Über dieses Protokoll werden i. d. R. alle für eine funktionierende Einwahlverbindung notwendigen Informationen ausgetauscht:

- Der Einwählende übergibt Kennung und Passwort,
- der Provider liefert nach vollzogener Autorisierung eine IP-Nummer
- und bei Bedarf auch Informationen über seine Nameserver zurück.

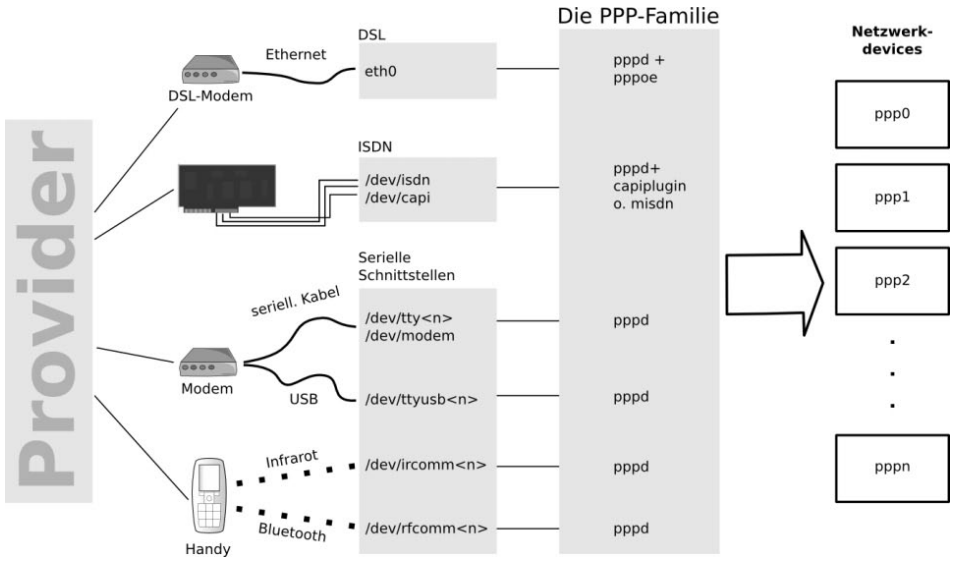


Abbildung 2.5: Einwahlmöglichkeiten mit PPP

Unter Linux erledigt der PPP-Dämon diese Aufgabe und stellt nach erfolgter Autorisierung im System eine Netzwerkschnittstelle `pppn` zur Verfügung, die sich unabhängig von der nach außen verwendeten Technik wie eine Standardnetzwerkschnittstelle verhält und auch so genutzt werden kann.

Die Aufgabe der Konfiguration des PPP-Dämons, die nicht immer trivial ist, übernehmen verschiedene Tools wie z. B. `kppp` oder `wvdial`. In der Regel finden Sie bei den Distributionen ebenfalls Tools, die Ihnen die Konfiguration der Einwahl abnehmen.

Wir wollen Ihnen am Beispiel der Einwahl über eine DSL-Verbindung die Funktionsweise des PPP-Dämon demonstrieren:

2.4.1 Ins Internet mit DSL

DSL steht je nach technischem Background für **D**igital **S**ubscriber **L**ine oder **D**igital **S**ubscriber **L**oop. Dabei handelt es sich um eine Technologie, die über das herkömmliche Kupfertelefonkabel deutlich mehr Daten übertragen kann als ein einfacher Sprachkanal, an den z. B. ein analoges Modem gebunden ist. Das sogenannte DSL-Modem wird dabei wegen der höheren Bandbreite nicht mehr über ein serielles Kabel, sondern direkt über ein Netzwerkkabel (i. d. R. Ethernet) angeschlossen. Für den Internet-Zugang über eine DSL-Verbindung wird unter Linux ebenfalls der PPP-Dämon genutzt, mit einem den technischen Rahmenbedingungen angepassten Protokoll namens PPPoE (**PPP o**ver **E**thernet).

Was benötigen Sie, um die DSL-Verbindung konfigurieren zu können?

- Eine vom Kernel erkannte Netzwerkkarte, die aber nicht konfiguriert sein sollte
- Ein betriebsbereites und angeschlossenes DSL-Modem, das mit dem PC verbunden ist und bei dem das Lämpchen »DSL-Link« grün leuchten sollte
- Die Zugangsdaten Ihres Providers, respektive Kennung und Passwort

Jede Distribution bietet Ihnen eine dialoggestützte Konfiguration, um Ihren DSL-Zugang einzurichten. Auf Debian-basierten Systemen können Sie das textbasierte Tool `pppoeconf` nutzen, unter Fedora steht Ihnen `adsl-setup` zur Verfügung während SUSE-Nutzer in YaST unter NETZWERK GERÄTE die DSL-Konfiguration verwenden können.

SUSE-Tipp

Unter SUSE müssen Sie zwischen Ethernet und ATM wählen. Im deutschsprachigen Raum werden aber i. d. R. DSL-Modems genutzt, die über eine Ethernet-Netzwerkkarte angesprochen werden. Wir beschränken uns daher hier auf die Konfiguration von PPPoE.

Was wird bei der Konfiguration von Ihnen erfragt? Wir gehen die Fragen einmal kurz am Beispiel des Debian-Scripts `pppoeconf` durch, das Sie auf der Kommandozeile mit root-Berechtigung starten sollten. Die Konfiguration wird in der Datei

/etc/ppp/peers/dsl-provider gespeichert und bei jedem Start von pppoeconf überschrieben.

- Als Erstes muss die Netzwerkkarte bekannt sein, an der das DSL-Modem angeschlossen ist. Das Script stellt daher zuerst eine Liste der in Frage kommenden Geräte zusammen.
- Sie haben im weiteren Verlauf die Möglichkeit, einige Optionen des PPP-Dämons zu verändern. Solange Sie nicht wissen, warum Sie dies tun sollten, können Sie getrost die angebotenen Vorgaben übernehmen. Wir werden Ihnen im Folgenden nur die wichtigen Angaben erläutern.
- Die vom Provider gelieferten Zugangsdaten, respektive Kennung und Passwort, werden i. d. R. über zwei Verfahren ausgetauscht: PAP (**P**assword **A**uthentication **P**rotocol) und CHAP (**C**hallenge **H**andshake **A**uthentication **P**rotocol). PAP ist die ältere Variante, in der das Passwort im Klartext übertragen wird; dagegen werden bei CHAP die Passwörter verschlüsselt übertragen. Mittlerweile ziehen die Provider daher eine CHAP- einer PAP-Autorisierung vor.
- Wenn Sie keine Flatrate gebucht haben oder sich mit einem Laptop einwählen, kann es sinnvoll sein, die Einwahl nicht bei jedem Start des Computers zu starten. Wir erklären Ihnen gleich, wie Sie die Einwahl auch manuell durchführen können.

Nach dem Aufruf finden Sie im Verzeichnis /etc/ppp/peers die Konfigurationsdatei:

Listing 2.5: dsl-provider

```
# Minimalistic default options file for DSL/PPPoE connections

noipdefault
defaultroute
replacedefaultroute
hide-password
#lcp-echo-interval 30
#lcp-echo-failure 4
noauth
persist
#mtu 1492
#usepeerdns
plugin rp-pppoe.so eth0
user "<Ihre Kennung>"
```

Ein kurzer Überblick über die wichtigsten Optionen:

- plugin zeigt an, auf welche Art der PPP-Dämon die Einwahl ausführen soll. Das Plugin rp-pppoe wird für die Einwahl über PPPoE benötigt.
- Mit user wird die Kennung angegeben, die für die Autorisierung notwendig ist. Das Passwort finden Sie in der Datei /etc/ppp/pap-secrets bzw. /etc/ppp/chap-secrets. Da für die Einwahl i. d. R. die automatische Auswahl von PAP oder

CHAP eingestellt ist, sollten in beiden Dateien (wenn die eine nicht auf die andere verlinkt ist) die Kennung/Passwort-Paare eingetragen sein.

- `defaultroute` und `replacedefaultroute` sagen dem PPP-Dämon, dass er nach der Einwahl den gesamten Verkehr über die Einwahlschnittstelle routen soll. Dies ist bei der Einwahl mit einem einzelnen PC die passende Einstellung; jedoch bei einem Server, der als Gateway fungieren soll, würden diese Optionen dafür sorgen, dass der Rechner das lokale Netzwerk nicht mehr findet.

Jetzt können Sie mit Hilfe der Kommandos `pon`, `poff` und `plog` Ihre DSL-Verbindung starten, beenden und überwachen:

```
hugo:~# pon dsl-provider
Plugin rp-pppoe.so loaded.
hugo:~# plog
localhost pppd[12930]: Using interface ppp0
localhost pppd[12930]: Connect: ppp0 <--> eth0
localhost pppd[12930]: CHAP authentication succeeded
localhost pppd[12930]: CHAP authentication succeeded
localhost pppd[12930]: peer from calling number nn:nn:nn:nn:nn:nn authorized
localhost pppd[12930]: Cannot determine ethernet address for proxy ARP
localhost pppd[12930]: local  IP address nnn.nnn.nnn.nnn
localhost pppd[12930]: remote IP address nnn.nnn.nnn.nnn
localhost pppd[12930]: primary   DNS address nnn.nnn.nnn.nnn
localhost pppd[12930]: secondary DNS address nnn.nnn.nnn.nnn
hugo:~#
```

Dem Script `pon` geben Sie als Option den Namen der gerade erstellten Konfigurationsdatei mit. Sie können verschiedene Konfigurationsdateien für verschiedene Provider führen. Das Kommando `plog` zeigt Ihnen einen Ausschnitt aus dem Syslog, der die Log-Einträge des PPP-Dämons enthält. Hier können Sie erkennen, dass die Authentisierung geklappt hat und wir eine IP-Adresse und zwei DNS-Server zugewiesen bekommen haben (die wir hier natürlich anonymisiert darstellen).

```
hugo:~# poff
hugo:~# plog
localhost pppd[12930]: Terminating on signal 15
localhost pppd[12930]: Connect time 9.8 minutes.
localhost pppd[12930]: Sent 294558 bytes, received 810253 bytes.
localhost pppd[12930]: Connection terminated.
localhost pppd[12930]: Exit.
hugo:~#
```

Mit dem Kommando `poff` können Sie die DSL-Verbindung wieder deaktivieren. Ein Aufruf ohne Optionen genügt, und – *puff* – ist sie weg. :-) Da Sie keine direkte Rückmeldung bekommen, können Sie noch ein `plog` nachschieben, das Ihnen die Beendigung der Verbindung bestätigt. Wie Sie im Beispiel sehen können, loggt der PPP-Dämon zu jeder Verbindung die Zeit und die übertragenen Bytes.

2.4.2 Modem-Einwahl mit KPPP

Für die Modem-Einwahl existiert mit KPPP ein auf der KDE-Oberfläche basierendes Tool, mit dem Sie alle Parameter für eine Modem-Einwahl steuern können.

Sie müssen im Prinzip drei Dinge konfigurieren:

1. **Die technischen Daten des Modems:** Auch wenn Ihnen die technischen Begriffe wenig sagen, gibt es hier nur wenige Fallstricke.
   - Das Device, über das das Modem angesprochen wird, wird von der Distribution auf `/dev/modem` verlinkt, wenn das Modem bei der Installation erkannt wurde. Von den Einträgen, die KPPP zur Auswahl anbietet, sind die Einträge `/dev/ttyS`*n* diejenigen, die Sie als Erstes ausprobieren sollten, wenn `/dev/modem` nicht funktioniert. Dabei entspricht `ttyS0` dem DOS/Windows-Device `com1` usw. Über den Reiter MODEM erreichen Sie zwei Funktionen TERMINAL und MODEM ABFRAGEN, mit denen Sie ausprobieren können, ob das Modem an dieser Schnittstelle angeschlossen ist.
   - FLUSSKONTROLLE und ZEILENENDE sind auf die am häufigsten genutzten Werte eingestellt. Hier sollten Sie zuletzt einen Fehler suchen.

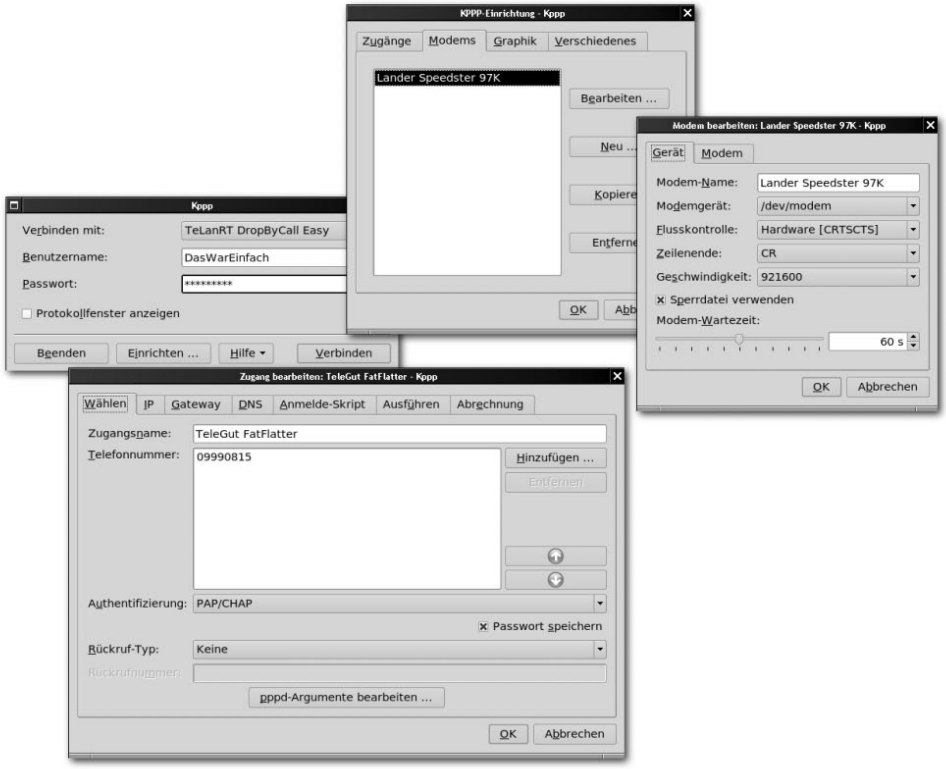


Abbildung 2.6: KPPP

- Die GESCHWINDIGKEIT bezieht sich an dieser Stelle nicht auf die Übertragungsgeschwindigkeit der Telefonverbindung, sondern auf die Geschwindigkeit, mit der Daten zwischen Modem und Computer übertragen werden. Hier sollte auf jeden Fall eine größere Geschwindigkeit eingetragen werden, als das Modem über die Telefonleitung übertragen kann. In der Regel können Sie den voreingestellten Wert beibehalten.

2. **Die Konfigurationsdaten des Providers:** Hier tragen Sie die Telefonnummer und einen aussagekräftigen Namen ein. Die Authentifizierung erfolgt, wenn vom Provider keine anderen Angaben vorliegen, über PAP bzw. CHAP. Alle weiteren Angaben wie IP-Nummer, Gateway, DNS und die weiteren Reiter können Sie für eine Standard-Einwahl getrost ignorieren.
3. **Kennung und Passwort** müssen Sie erst bei der Einwahl eingeben. Sie wählen dort einen der konfigurierten Provider aus und geben dazu dann die passende Kennung und respektive das Passwort an. Wenn Sie bei der Einrichtung der Provider-Konfiguration das Häkchen bei PASSWORT SPEICHERN angeschaltet haben, merkt sich KPPP die Kennung/Passwort-Kombination für das nächste mal.

Danach erledigt KPPP den Rest für Sie. Dazu gehören alle Dinge, die normalerweise ein Systemadministrator konfigurieren müsste, wie z. B. das Setzen der IP-Nummer, der dazugehörigen Routen und die Anpassung des Nameservice. Während der Verbindung können Sie KPPP beibringen, im Tray des KPanels zu verschwinden. Außerdem können Sie die Kostenregeln der Provider in KPPP eintragen und erhalten so einen ungefähren Überblick, welche Kosten Ihre Aufenthalte im Internet bisher verursacht haben.

2.4.3 Automatisierte Einwahl mit wvdial

Es gibt Fälle, in denen ein Einwahlprogramm mit grafischer Oberfläche nicht zu gebrauchen ist, beispielsweise wenn (auf einem Server) nur die Textoberfläche zur Verfügung steht, wenn die Einwahl scriptgesteuert ausgeführt werden soll oder wenn der Benutzer einfach lieber tippt als klickt. Für diese Fälle bietet sich das Programm `wvdial` an, das auf der Kommandozeile gestartet wird, seine Konfiguration im Regelfall aus der Datei `/etc/wvdial.conf` liest und alsdann versucht, das Modem zur Einwahl zu veranlassen.

Wenn Sie `wvdial` benutzen wollen, brauchen Sie die Konfigurationsdatei nicht mühsam selbst zu tippen. Im Paket `wvdial` findet sich das Programm `wvdialconf`, das Sie (mit root-Rechten) starten können, um eine Standardkonfiguration zu erzeugen. Es sucht selbsttätig im System nach einem ansprechbaren Modem (ein eventuell notwendiges Kernelmodul muss also zuvor geladen worden sein). Dann experimentiert es ein wenig mit den halbwegs standardisierten Steuerkommandos (»AT-Strings«) herum, bis es eine brauchbare Initialisierung herausgefunden hat, und schreibt eine Musterkonfiguration in die Datei `/etc/wvdial.conf`:

```
hugo:~# wvdialconf
Editing '/etc/wvdial.conf'.
```

```
Scanning your serial ports for a modem.

ttyS0<*1>: ATQ0 V1 E1 -- failed with 2400 baud, next try: 9600 baud
ttyS0<*1>: ATQ0 V1 E1 -- failed with 9600 baud, next try: 115200 baud
ttyS0<*1>: ATQ0 V1 E1 -- and failed too at 115200, giving up.
Modem Port Scan<*1>: S1   S2   S3
WvModem<*1>: Cannot get information for serial port.
ttySL0<*1>: ATQ0 V1 E1 -- OK
ttySL0<*1>: ATQ0 V1 E1 Z -- OK
ttySL0<*1>: ATQ0 V1 E1 S0=0 -- OK
ttySL0<*1>: ATQ0 V1 E1 S0=0 &C1 -- OK
ttySL0<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 -- OK
ttySL0<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0 -- OK
ttySL0<*1>: Modem Identifier: ATI -- SmartLink Soft Modem
ttySL0<*1>: Speed 4800: AT -- OK
ttySL0<*1>: Speed 9600: AT -- OK
ttySL0<*1>: Speed 19200: AT -- OK
ttySL0<*1>: Speed 38400: AT -- OK
ttySL0<*1>: Speed 57600: AT -- OK
ttySL0<*1>: Speed 115200: AT -- OK
ttySL0<*1>: Speed 230400: AT -- OK
ttySL0<*1>: Speed 460800: AT -- OK
ttySL0<*1>: Max speed is 460800; that should be safe.
ttySL0<*1>: ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0 -- OK

Found a modem on /dev/ttySL0.
/etc/wvdial.conf: Can't open '/etc/wvdial.conf' for reading: No such file or directory
/etc/wvdial.conf: ...starting with blank configuration.
Modem configuration written to /etc/wvdial.conf.
ttySL0: Speed 460800; init "ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0"
hugo:~# cat /etc/wvdial.conf

[Dialer Defaults]
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
Modem Type = Analog Modem
; Phone = <Target Phone Number>
ISDN = 0
; Username = <Your Login Name>
Init1 = ATZ
; Password = <Your Password>
Modem = /dev/ttySL0
Baud = 460800
hugo:~#
```

Gerätename und Kenndaten des Modems sowie der Initialisierungsstring sind korrekt in die Datei eingetragen. Sie müssen nur noch in drei Zeilen jeweils das

Semikolon als Kommentarzeichen entfernen und die von Ihrem Provider vergebenen Zugangsdaten (Telefonnummer, Login-Kennung und Passwort) anstelle der Platzhalter einfügen. Anschließend können Sie `wvdial` (ohne Parameter) aufrufen – mit etwas Glück sollte die Modem-Einwahl jetzt reibungslos ablaufen.

Weil die Einwahl Kosten verursacht, kann sie nicht von jedem Benutzer ohne weiteres gestartet werden. Wenn die Modem-Gerätedatei (hier `/dev/ttySL0`) Schreibrechte für eine besondere Gruppe (z. B. `dialout`) aufweist, kann auch ein Normalbenutzer, der dieser Gruppe zugeordnet ist, mit `wvdial` das Modem benutzen.

Das Programm bleibt so lange aktiv, wie die Modem-Verbindung besteht; wenn das Modem auflegen soll, muss `wvdial` (z. B. durch Strg + C) beendet werden. Falls längere Zeit keine Daten übertragen werden, bricht `wvdial` die Verbindung nach einer einstellbaren Zeitspanne (Option `Idle Seconds`) ab. Verlassen Sie sich aber nicht auf diese Automatik, denn wenn von einer Anwendung im Hintergrund regelmäßig Datenpakete versandt werden, bleibt die Verbindung endlos bestehen und beschert Ihrem Telefonanbieter ebenso endlose Einnahmen.

Steht Ihnen ein weiterer Zugangsanbieter zur Wahl, so können Sie der Konfigurationsdatei einen weiteren Abschnitt hinzufügen (zusätzlich zu dem mit `Dialer Defaults` gekennzeichneten), dessen Name beim Aufruf von `wvdial` als Kommandozeilenoption angegeben werden kann.

2.4.4 Multifunktional: ISDN

Um dem Thema ISDN gerecht zu werden, müsste man alleine ein ganzes Buch schreiben. Wir wollen an dieser Stelle nur kurz auf das Thema »Einwahl« eingehen. Glücklicherweise ist die Unterstützung von ISDN sehr gut, auch Dank der Förderung des Projekts *isdn4linux* durch SUSE in der Vergangenheit. Die meisten passiven ISDN-Karten werden vom Kernel erkannt. Außerdem bieten die meisten Distributionen Tools für die Konfiguration des Internetzugangs per ISDN. Insgesamt ist das Paket *isdn4linux* ein mächtiges Werkzeug, dessen Konfiguration aber nicht immer einfach ist.

ISDN-Karten werden unter Linux als Netzwerkkarten mit speziellen Fähigkeiten behandelt. Die Einrichtung einer ISDN-Karte ist aber selten ein Problem, weil

- das zur ISDN-Karte passende Kernelmodul bei den von der Distribution mitgelieferten Kerneln dabei ist und bei der Installation automatisch erkannt werden sollte,
- unter Umständen von aktiven Karten benötige Firmware als `rpm`- bzw. `deb`-Paket bereitsteht und automatisch geladen wird.

Wir wollen auf die unterschiedlichen Protokolle, die ISDN bietet, hier nicht näher eingehen, sondern Ihnen die für den deutschen Standard 1TR1 notwendigen Konfigurationsdaten kurz erläutern:

- Um sich irgendwo einwählen zu können, muss Ihr Anschluss mit der richtigen Nummer konfiguriert sein, der MSN. Wenn Sie einen Internetanschluss mit mehreren Nummern besitzen, können Sie die Nummer, die von außen als Ihre

MSN konfiguriert ist, mit Hilfe eines Anrufs erkennen. Wenn die Karte richtig erkannt und geladen worden ist, sollten Sie in `/var/log/syslog` den Anrufsversuch protokolliert bekommen, inklusive der eigenen MSN in internationaler Notation (also mit Länderkennung und der Vorwahl ohne führende 0):

```
[...] Call to tei 127 from [...], Germany cellphone [...]
                                                    on +49 nnn/nnnnnnn RING (Speech)
[...] Call to tei 127 from [...], Germany cellphone [...]
                                                  on +49 nnn/nnnnnnn HLC: CCITT, Telefonie
[...] Call to tei 66 from [...], Germany cellphone [...] on +49 nnn/nnnnnnn HANGUP
```

- Sie brauchen die Nummer des Providers, bei dem Sie sich einwählen wollen,
- mit der dazugehörigen Kennung und dem Passwort. Fallstricke gibt es an dieser Stelle nur, wenn Sie Ihren ISDN-Anschluss von einem Anbieter gemietet haben, der keine Call-by-Call-Angebote anderer Anbieter zulässt. Der nicht funktionierende Verbindungsaufbau ist dann nicht Ihrem System, sondern dem Anbieter anzulasten.

Die Konfigurationen werden i. d. R. direkt einer System-Schnittstelle zugeordnet, die bei ISDN nicht `ppp0`, sondern `ippp0` genannt wird; dies liegt u. a. auch daran, dass von *isdn4linux* eine speziell angepasste Version des PPP-Dämons mitgeliefert wird, der `ipppd` heisst. Zwei Konfigurationsdateien werden im Verzeichnis `/etc/isdn` angelegt: ein ausführbares Script für den Device mit dem Namen `device.ippp0` und eine Konfigurationsdatei für den `ipppd` mit dem Namen `ipppd.ippp0`, die jeweils als Endung den Namen der Schnittstelle tragen.

Für eine Verbindung werden durch ein Init-Script der `ipppd` und ein Log-Dämon `isdnlog` gestartet. Der `ipppd` kann dann durch das Kommando `isdnctrl` gesteuert werden. Wird der ISDN-Service beim Start automatisch hochgefahren, können Sie durch die Option `DIALMODE=manual` in der Datei `device.ippp0` den automatischen Verbindunsaufbau abschalten. In diesem Fall wird das System beim Booten des Rechners zwar hochgefahren, aber die Verbindung erst nach `isdnctrl dial ippp0` gestartet. Damit haben Sie eine bessere Kontrolle über Aufbau und Abbruch der Wählverbindung.

Funktioniert alles, bekommen Sie in wenigen Sekunden eine Verbindung. Der Verbindungsaufbau funktioniert dabei i. d. R. so schnell, dass Sie das System auf ein Dial-On-Demand konfigurieren können, ohne bei der normalen Nutzung die Verzögerungen zu bemerken, die durch den Einwahlvorgang entstehen.

Weitere Hinweise zu ISDN und den Möglichkeiten, die eine ISDN-Karte zusätzlich zur reinen Internetnutzung noch bietet, finden Sie im unten genannten HOWTO. Eine Software, die ebenfalls ISDN nutzen kann, ist Asterisk. Damit können Sie eine Telefonanlage auf der Basis von Voice-Over-IP einrichten und mit einer oder mehreren ISDN-Karten ein Gateway in das Festnetz schaffen.

Weitere Literatur

- Bereits etwas älter, aber immer noch gültig – das »ISDN-Howto«, `http://www.klaus.franken.de/DE-ISDN-HOWTO/`
- Asterisk, `http://www.asterisk.org/`

3 Elementare Dienste im Netzwerk

Bisher haben wir uns hauptsächlich damit beschäftigt, was Sie tun müssen, um Ihr Linux-System ans Netzwerk anzuschließen. In diesem Kapitel widmen wir uns jetzt einigen grundlegenden Diensten, die speziell in Unix/Linux-Umgebungen existieren und die Sie von anderen Betriebssystemen in dieser Form vielleicht nicht kennen.

3.1 Mit X im Netz

Netzwerkfähigkeiten sind in X auf einer so grundlegenden Ebene realisiert, dass ohne zusätzliche Software ein Datenaustausch zwischen verschiedenen Rechnern stattfinden kann. Dies stellt jedoch gleichzeitig das Problem dar: Eine Modernisierung des Client-Server-Modells von X ist bisher daran gescheitert, dass die dazu notwendigen Funktionen so tief im X-System verankert sind, dass Veränderungen hier große Auswirkungen auf die Abwärtskompatibilität hätten. Daher findet der Datenaustausch zwischen X-Server und X-Client weiterhin ohne zeitgemäße Verschlüsselung und Datenkomprimierung statt, was das Arbeiten über netzwerktechnisch weite Strecken mitunter stark behindert. Trotzdem stellen die Netzwerkfähigkeiten von X im internen Netzwerk immer noch eine einzigartige Lösung dar, die in anderen Systemen in dieser Form nur durch mitunter enormen zusätzlichen Softwareaufwand zu realisieren ist.

Lassen Sie uns aber kurz die grundlegenden Einstellungen rekapitulieren, die für die Kontaktaufnahme zwischen X-Server und -Client notwendig sind: Zuallererst sollten wir uns vergegenwärtigen, dass der X-Server auf dem Rechner läuft, auf dem sich die Grafikkarte befindet; in einem X-Terminal-vs.-Server-Verhältnis läuft der X-Server also auf dem Client und die X-Client-Software auf dem Server. Aber wenn Sie im Hinterkopf behalten, dass ein X-Server ohne Grafikhardware ungefähr so nützlich ist wie ein Auto ohne Motor, dann werden Sie auch bei diesen komplizierten Beziehungsverhältnissen den Überblick behalten[1].

[1] Wie immer ist dies auch eine eigentlich unzulässige Vereinfachung, da es auch X-Server gibt, die ohne Grafikkarte laufen. Aber erstens ist dies kein X-Buch und zweitens wollen wir nicht, dass Sie das Buch jetzt schon verzweifelt beiseitelegen.

Starten Sie einen X-Server auf Ihrem lokalen Linux-Rechner, läuft dieser i. d. R. unter der Nummer »0«. Dies wird in einer Umgebungsvariable verewigt, die den Namen DISPLAY\index{DISPLAY} trägt und den Wert :0 enthalten sollte[2]. Diese Information wird von den verschiedenen X-Clients ausgewertet und für den Zugriff auf den X-Server genutzt. Sollten Sie also mehrere X-Server auf Ihrem Rechner laufen lassen, weil Sie z. B. eine Dual-Head-Grafikkarte Ihr Eigen nennen und an dieser zwei Bildschirme angeschlossen sind, können Sie durch eine Veränderung dieser Umgebungsvariable die Clients auf die unterschiedlichen Monitore schicken[3].

Damit jetzt nicht Hinz und Kunz auf Ihrem X-Server Fenster ausgeben können, existiert noch ein weiterer Mechanismus, mit dem sich die X-Clients beim Server autorisieren müssen. Das geschieht heutzutage i. d. R. durch ein spezielles Cookie, das von einem Programm namens xauth erzeugt und verwaltet wird. Dieses wird in der Datei .Xauthority im Home-Verzeichnis Ihres Benutzerkontos abgelegt. Jedes X-Programm, das Sie unter Ihrer Kennung starten, wird aus dieser Datei das für den X-Server passende Cookie laden und sich damit autorisieren. Nur wenn dies mit dem vom X-Server verwalteten Wert übereinstimmt, wird eine grafische Ausgabe erlaubt.

```
victor@hugo:~$ echo $DISPLAY
:0
victor@hugo:~$ xauth list
hugo:0  MIT-MAGIC-COOKIE-1  61a0e9b797f9e5770865b2b4ae778544
victor@hugo:~$
```

Sie werden die Problematik sofort erfassen, wenn Sie eine Shell öffnen und dort das Kommando su - eingeben. Bei einem Aufruf dieser Form werden die Informationen über die Kennung, unter der Sie den X-Server gestartet haben, über Bord geworfen. Daher können ohne eine Anpassung der Umgebungsvariablen keine Programme auf der grafischen Oberfläche ausgegeben werden. Befindet man sich auf dem Rechner, auf dem die Datei mit den magischen Cookies liegt, kann man den X-Clients mit Hilfe der Umgebungsvariable XAUTHORITY eine Autorisierung ermöglichen:

```
victor@hugo:~$ su -
Password:
hugo:~# xclock
Error: Can't open display:
hugo:~# echo $DISPLAY
hugo:~# export DISPLAY=:0
hugo:~# xclock
Xlib: connection to ":0.0" refused by server
Xlib: No protocol specified
```

[2] Den Doppelpunkt kriegen wir später. :-)

[3] Das geht übrigens auch mit nur einer Grafikkarte. Sie können ja einmal auf eine Textkonsole umschalten und dort das Kommando X :1 aufrufen. Nett nicht? Sie erinnern sich auch noch daran, dass Sie mit STRG + ALT + BACKSPACE den X-Server wieder beenden können?

```
Error: Can't open display: :0
hugo:~# export XAUTHORITY=~victor/.Xauthority
hugo:~# xclock &
hugo:~#
```

Anstatt auf die Datei mit dem magischen Cookie zu verweisen, kann man dieses auch in der lokalen Xauthority-Datei des Nutzers abspeichern. Damit funktioniert der Zugriff dann ebenfalls. Wir demonstrieren Ihnen das noch einmal an einem lokalen Wechsel zur root-Kennung:

```
victor@hugo:~$ xauth list
hugo:0  MIT-MAGIC-COOKIE-1  61a0e9b797f9e5770865b2b4ae778544
victor@hugo:~$ su -
Password:
hugo:~# xauth list
xauth:  creating new authority file /root/.Xauthority
hugo:~# xauth add hugo:0  MIT-MAGIC-COOKIE-1  61a0e9b797f9e5770865b2b4ae778544
hugo:~# export DISPLAY=hugo:0
hugo:~# xclock &
hugo:~#
```

Praktischerweise hat die Ausgabe des Kommandos `xauth list` das Format, das Sie benötigen, um mit Hilfe von `xauth add ...` einen Eintrag einer Xauthority-Datei hinzuzufügen. Da bei jedem Start aber ein neues Cookie generiert wird, funktioniert diese Methode nur während der aktuell laufenden Sitzung.

Wir haben bisher immer lokal agiert, aber unser letztes Code-Beispiel gibt bereits einen kleinen Hinweis, wie wir X-Clients über das Netzwerk auf einen anderen Rechner schicken. Sie können vor dem Doppelpunkt in der `DISPLAY`-Variable einen Hostnamen oder eine IP-Adresse angeben. Wenn der X-Client dort eine Ausgabe erzeugen darf, wird das Fenster auf dem Monitor dieses Rechners erscheinen.

Achtung

Bei den meisten Distributionen ist der Aufruf des X-Servers mittlerweile so konfiguriert, dass das System keine Zugriffe über das Netzwerk annimmt. Dies hat keinerlei Auswirkung auf den Remote-Zugriff mit Hilfe von `ssh` und die darüber »verschickten« grafischen Ausgaben. Der einfache Netzzugriff wird darüber jedoch abgeklemmt. Davon, wie Sie Ihre Session starten, hängt es nun ab, an welcher Stelle Sie dem X-Server das Lauschen auf Netzwerkzugriffe erlauben wollen:

- **KDM und XDM**: Im Konfigurationsverzeichnis dieser Display-Manager finden Sie eine Datei `Xservers`, in der Aufrufe des X-Servers abgelegt sind inklusive der Kommandozeilen-Optionen. Hier

müssen Sie die Option `-nolisten tcp` entfernen, damit der X-Server auch auf Anfragen über das Netzwerk hört.

- **GDM**: In der Datei `gdm.conf` finden Sie einen Eintrag `DisallowTCP=True`. Diesen müssen Sie auf `False` setzen, damit der vom GDM gestartete X-Server auch Netzwerkzugriffe zulässt.
- **Auruf über die Kommandozeile**: Für den Start von X existiert ein Kommando mit dem Namen `startx`. Dieses nutzt die Funktionen von `xinit`, um eine vollwertige X-Session zu starten. Die Konfigurationsoptionen für den Aufruf des X-Servers finden sich in der Datei `xserverrc` des Konfigurationsverzeichnisses von `xinit`. Dort muss wie oben die Option `-nolisten tcp` entfernt werden, um einen direkten Netzwerkzugriff zu erlauben.

Bevor Sie den Netzwerkzugriff dauerhaft erlauben, sollten Sie sich aber noch unseren Tipp in Abschnitt 11.3.5 durchlesen. Die Sicherheitsmechanismen von X sind modernen Anforderungen nicht immer gewachsen.

Neben der Cookie-Verwaltung gibt es eine weitere Möglichkeit, Zugriffskontrollen auf den X-Server einzurichten und zu verwalten. Mit Hilfe des Kommandos `xhost` können Sie Regeln für Hosts festlegen und diesen einen Zugriff erlauben.

```
knoppix@knoppix:~$ /sbin/ifconfig eth0
eth0      Protokoll:Ethernet  Hardware Adresse 00:A1:B0:09:48:E2
          inet Adresse:192.168.1.11  Bcast:192.168.1.255  Maske:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5175 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6047 errors:0 dropped:0 overruns:1 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:2696235 (2.5 MiB)  TX bytes:804348 (785.4 KiB)
          Interrupt:18 Basisadresse:0xa400
knoppix@knoppix:~$ xhost
access control enabled, only authorized clients can connect
knoppix@knoppix:~$ xhost add +192.168.1.10
knoppix@knoppix:~$ xhost
access control enabled, only authorized clients can connect
INET:192.168.1.10
knoppix@knoppix:~$
```

Ein Aufruf des Kommandos `xhost` listet die zurzeit freigegebenen Rechner auf. Mit `xhost add +192.168.1.10` haben wir jetzt auf unserem Knoppix-Testrechner die IP-Nummer des Servers für den Zugriff freigegeben. Wichtig ist das Plus-Zeichen vor der Angabe des Rechners; damit geben Sie an, dass der Rechner Zugriff erhalten soll. Mit einem Minus-Zeichen können Sie eine Freigabe wieder zurückziehen. Jetzt können wir auf unserem Server die `DISPLAY`-Variable setzen und dann X-Programme auf dem Knoppix-System ausgeben.

```
victor@hugo:~$ export DISPLAY=192.168.1.11:0
victor@hugo:~$ xclock &
victor@hugo:~$
```

Zu diesem Vorgehen seien jedoch noch einige Bemerkungen erlaubt:

- Mit `xhost` erlauben Sie über die IP-Nummer bzw. den Rechnernamen *jedem* Nutzer dieses Computers den Zugriff auf den X-Server. Eine Differenzierung zwischen verschiedenen Kennungen ist nicht mehr möglich.
- Außerdem werden die Darstellungsinformationen für Fenster und deren Inhalte nicht geschützt, also verschlüsselt, übertragen. Für die Server-Wartung sind also die Möglichkeiten vorzuziehen, die `ssh` bietet. Durch die Tunnelung der X-Übertragung ist hierbei kein Abhören möglich.
- Trotzdem kann `xhost` ein nützliches Feature sein: Haben Sie z. B. ein X-Terminal, das Sie für ein einfaches Kiosk-System nutzen wollen, können Sie darauf den Server freigeben, von dem aus eine grafische Ausgabe erfolgen soll. Damit können Sie ohne großen Aufwand von einem entsprechend geschützten Server aus über das Netzwerk grafische Ausgaben erzeugen wie z. B. Monitorprogramme, die die aktuelle Systemauslastung darstellen, oder einen Webbrowser, der die aktuellen Busverbindungen ausgibt.
- `xhost` kann aber auch eine Falle sein: Sie sollten auf keinen Fall das Kommando `xhost +` ausführen! Schon gar nicht, wenn Ihnen die Kollegen weismachen wollen, dass dabei ja gar nichts passieren könnte, da Sie ja keinen Rechner angegeben haben. Probieren Sie es einmal heimlich alleine und ungesehen aus. Und erinnern Sie sich daran, dass es auch noch ein `xhost -` gibt!

3.1.1 Mit X-Terminals am Server anmelden

Eine verblüffend einfache Möglichkeit stellt der Zugriff eines X-Servers auf einen im Netzwerk befindlichen Display-Manager dar. Als Rechnerleistung noch hauptsächlich von großen Servern bereitgestellt wurde und der Zugriff i. d. R. über Terminals erfolgte, wurden auch für grafische Oberflächen geeignete Geräte gebaut, die sogenannten X-Terminals. Diese beinhalteten im Prinzip nur eine Grafikkarte, eine Netzwerkkarte (oder eine anders geartete Möglichkeit des Remote-Zugriffs) und die für den Netzwerkzugriff und die grafische Darstellung notwendige Software. Bei einer Session wurde die gesamte vom Nutzer gestartete Software auf dem Server ausgeführt. Nur die Ausgabe landete auf dem X-Terminal.

Die Nachfolger der X-Terminals sind die heute als Thin-Clients bezeichneten Geräte. Darunter wird so ziemlich alles subsumiert, was möglichst klein ist, wenig kostet und wenig tut. Wenn Sie mit alter Hardware gesegnet sind und überlegen, wie Sie mit wenig Geld möglichst viel erreichen können, ist diese antike Art des vernetzten Arbeitens vielleicht eine Option. In diesem Fall müssen Sie nur darüber nachdenken, ob die Fähigkeiten der alten PCs ausreichen, eine vernünftige grafische Ausgabe bezüglich Auflösung und Farbtiefe zu erzeugen. Im späteren Betrieb müssen die behäbig arbeitenden Prozessoren dann nur noch Pixel verwalten. Wenn auch die

Monitore noch vernünftig arbeiten, wäre nur die Anschaffung eines passend ausgestatteten Servers notwendig, weil dieser die gesamte Rechenarbeit übernehmen muss.

Wir wollen an dieser Stelle keine Anleitung geben, wie Sie aus einem PC ein X-Terminal machen. Hier gibt es bereits Distributionen, die Ihnen diese Aufgabe abnehmen. Je nach Hardware sollten Sie testen, welche mit dem geringsten Aufwand bei Ihnen einzusetzen sind. Da alle Distributionen das Booten über das Netzwerk bzw. über Diskette oder CD erlauben und eine Installation für einen ersten Test nicht notwendig ist, sollte der Test auch nicht unnötig Mühe machen.

Linux-Distributionen, die für X-Terminals geeignet sind

- Linux Terminal Server Project: `http://www.ltsp.org/`
- ThinStation: `http://thinstation.sourceforge.net/`
- ThinTUX: `http://thintux.sourceforge.net/`

Was wir Ihnen aber zeigen und erklären wollen, ist der Zugriff auf einen im Netzwerk laufenden Display-Manager mit Hilfe eines X-Servers. Dazu ist notwendig:

- ein Server oder irgendwie gearteter PC mit Linux und einem laufenden Display-Manager. Wir werden in unserem Beispiel KDM benutzen. Das Prinzip ist aber bei allen anderen Display-Managern gleich. Es muss übrigens kein Bildschirm am Rechner angeschlossen sein. Der Display-Manager kann auch ohne ein X auf Anforderungen warten.
- ein Client. Dieser sollte in unserem Fall im Textmodus laufen, da wir später den X-Server per Hand starten wollen. Wenn nicht, ist auch egal; Sie können das Ganze auch mit einem zweiten X-Server ausprobieren.
- ein Netzwerk, in dem sich beide Rechner befinden und das sich nicht gerade in einem Hochsicherheitsbereich befindet. Beide Rechner sollten sich »sehen« können, und am einfachsten ist es, wenn sich keine Hindernisse dazwischen befinden wie z. B. eine Firewall.

Der Informationsaustausch zwischen Display-Manager und X-Server findet über das Protokoll XDMCP, das **X** **D**isplay **M**anager **C**ontrol **P**rotocol statt. Dieses ist aus Sicherheitsgründen bei allen aktuellen Distributionen deaktiviert. KDM besitzt in der Konfigurationsdatei `kdmrc` (meistens ganz am Ende der Datei) eine Sektion `[Xdmcp]`. Dort muss der Eintrag `Enable` auf `True` gesetzt werden:

Listing 3.1: kdmrc

```
[Xdmcp]
Enable=True
Willing=/etc/kde3/kdm/Xwilling
```

Diejenigen, die KDM auf einem bildschirmlosen Server laufen lassen, können in der Datei `Xservers` (i. d. R. im selben Verzeichnis wie die Datei `kdmrc`) alle zu startenden X-Server auskommentieren.

Listing 3.2: Xservers

```
# Xservers - local X-server list
...

#:0 local@tty1 /usr/X11R6/bin/X -nolisten tcp :0
#:1 local@tty2 reserve /usr/X11R6/bin/X -nolisten tcp :1
#:2 local@tty3 reserve /usr/X11R6/bin/X -nolisten tcp :2
```

Man kann (und muss) dem Display-Manager mitteilen, welche Rechner berechtigt sind, seinen Service zu nutzen. Dazu existiert im gleichen Verzeichnis eine Datei `Xaccess`. Dort können wir für den Anfang die Zeile mit dem * auskommentieren, damit für die Testphase jede Anfrage angenommen wird:

Listing 3.3: Xaccess

```
# Xaccess - Access control file for XDMCP connections
#
...
*                          #any host can get a login window
...
```

Jetzt ist es an der Zeit, den Display-Manager zu starten (oder neu zu starten):

```
hugo:~# /etc/init.d/kdm restart
hugo:~#
```

Zusammengefasst müssen zwei Konfigurationsdateien angefasst werden:

1. Sie müssen XDMCP in der Datei `kdmrc` aktivieren, und
2. Sie müssen Host-Adressen in der Datei `Xaccess` eintragen, denen der Zugriff erlaubt ist.

Tipp

In der Datei `Xaccess` können Sie Hostnamen über Platzhalter angeben, z. B. mit »*.beispiel.org«. Die Platzhalter funktionieren nur leider nicht bei IP-Nummern.

Wenn der Zugriff trotzdem nicht funktioniert, müssen wir kontrollieren, ob wir uns eventuell durch Firewall-Einstellungen den Zugriff von außen abgeklemmt haben. Ein Blick in die Datei `/etc/services` zeigt uns, welche Ports für XDMCP und den Zugriff auf einen X-Server über das Netzwerk freigeschaltet sein müssen:

```
victor@hugo:~$ fgrep xdm /etc/services
xdmcp           177/tcp                         # X Display Mgr. Control Proto
xdmcp           177/udp
victor@hugo:~$ fgrep x11 /etc/services
x11             6000/tcp        x11-0           # X Window System
x11             6000/udp        x11-0
x11-1           6001/tcp
x11-1           6001/udp
x11-2           6002/tcp
x11-2           6002/udp
x11-3           6003/tcp
x11-3           6003/udp
x11-4           6004/tcp
x11-4           6004/udp
x11-5           6005/tcp
x11-5           6005/udp
x11-6           6006/tcp
x11-6           6006/udp
x11-7           6007/tcp
x11-7           6007/udp
victor@hugo:~$
```

Das XDMCP-Protokoll nutzt die Ports 177, während die Netzwerk-Zugriffe auf einen X-Server über die Portnummer 6000 + Display-Nummer laufen. Wenn Sie es sich leisten können, schalten Sie für die Tests auf beiden Rechnern die Firewall ab; wenn nicht, müssen Sie die Portnummern für XDMCP und den von Ihnen gestarteten X-Server freigeben; bitte sowohl tcp als auch udp[4].

Damit wechseln wir auf den Client und versuchen dort einmal, uns ein Login von unserem Server zu besorgen. Wechseln Sie auf eine Textkonsole (wenn Sie nicht schon eine solche vor sich haben) und melden Sie sich dort mit einer normalen Kennung an. Alles am Netz? Kabel drin? Helm auf? X auf dem Client konfiguriert? Dann wollen wir mal:

```
victor@theodor:~$ X -broadcast

Fatal server error:
Server is already active for display 0
        If this server is no longer running, remove /tmp/.X0-lock
        and start again.

When reporting a problem related to a server crash, please send
the full server output, not just the last messages.
```

[4] Sie können das übrigens auch alles auf einem Rechner ausprobieren. Aber das wäre doch langweilig, oder?

```
Please report problems to submit@bugs.debian.org.
# Ach, Sie haben schon einen X-Server laufen?
victor@theodor:~$  X :1 -broadcast
```

Sollte bei ersten Aufruf eine Fehlermeldung erscheinen, versuchen Sie es mal mit dem zweiten. In diesem Fall läuft bereits ein X-Server auf Ihrem Client und wir starten den zweiten einfach auf einem anderen Display. Danach sollte die grafische Ausgabe starten und der Display-Manager des Servers auf Ihrem Monitor erscheinen. Jetzt können Sie sich mit einer Nutzer-Kennung anmelden, die auf dem Server gültig ist. Nach der Anmeldung wird dann auf dem Server die Desktop-Umgebung gestartet.

Next-Generation X: die NX-Clients und Server. Wenn Sie, egal ob mit Hilfe von SSH oder auf anderem Weg, den grafischen Remote-Zugriff auch einmal mit Netzwerkverbindungen mit geringer Bandbreite ausprobiert haben, werden Ihnen vor allem in puncto Geschwindigkeit die Nachteile dieser Art der Übertragung aufgefallen sein. Um insbesondere diese Problematik zu lösen ist, die Firma NoMachine mit ihren NX-Produkten angetreten. Teile der Software sind dabei unter der GPL veröffentlicht, sodass bereits auch mit *FreeNX* ein freier NX-Server existiert. NX-Clients werden von NoMachine kostenlos zur Verfügung gestellt.

3.2 Remote-Konfiguration und Administration mit SSH

Die Nutzung einer Kommandozeile auf Servern im Netzwerk ist so alt wie das Internet selbst: in den Anfängen erzählte man den Rechnern mit Hilfe von **Telnet** (Port 23) oder der **Remote Shell**, was sie zu tun und zu lassen hatten. Mit dem Wachstum des Internet und der Bedeutung der Server für Nutzer mit guter und vor allem wegen derjenigen mit böser Absicht wuchsen die Anforderungen an einen durch Kryptografie geschützten Zugriff, bei dem die übertragenen Daten nicht von dritter Seite belauscht werden konnten. Das Resultat **ssh** (Port 22) war neben den recht komplizierten Versuchen mit Kerberos ein einfach zu nutzender Client-Server-Ansatz als Ersatz für `telnet` bzw. `rsh`. Als die ssh-Entwickler ihr Produkt zu Geld machen wollten und in ein proprietäres Tool umwandelten, wurde eine freie Variante entwickelt, die – aufgespalten in die Projekte **openssl** und **openssh** – mittlerweile in allen Linux-Distributionen enthalten sind.

Secure Shell	Debian	Fedora	SUSE	Windows
Secure Shell-Client Die Client-Software, um über das Netzwerk auf andere Server zuzugreifen	ssh	openssh-clients	openssh	Putty OpenSSH f. Windows
Secure Shell-Server Die Server-Software, um Zugriff von außen zu ermöglichen	ssh	openssh-server	openssh	OpenSSH f. Windows

Tabelle 3.1: Paketübersicht

Secure Shell	Debian	Fedora	SUSE	Windows
SSH-Askpass Tool, das ssh die Nachfrage nach einem Passwort unter X ermöglicht	ssh-askpass ssh-askpass-gnome	openssh-askpass gaskpass	openssh-askpass openssh-askpass-gnome	—

Tabelle 3.1: Paketübersicht (Fortsetzung)

Wir werden Ihnen hier die grundlegenden Funktionen von SSH vorstellen. Man sollte die Secure Shell jedoch nicht unterschätzen: Hier sind mit wenigen Funktionen eine ganze Reihe von Einsatzmöglichkeiten vorhanden, die nicht auf den ersten Blick ersichtlich sind; dazu mehr in Abschnitt 11.2.3 auf S. 277.

3.2.1 Aktivierung des SSH-Servers

Da SSH zur Standardsoftware einer Linux-Installation gehört, müssen Sie sich i. d. R. nicht mehr um die Installation der entsprechenden Pakete kümmern. Während der Installation werden auch alle notwendigen Vorkehrungen getroffen, um den SSH-Server starten zu können, wie z. B. die Generierung der notwendigen kryptografischen Schlüssel. Der SSH-Server läuft als separater Prozess und wird nicht über den `inetd` (siehe auch Abschnitt 11.3.3) gestartet. Daher müssen Sie noch dafür sorgen, dass der Prozess beim Booten automatisch aufgerufen wird.

Debian	Am einfachsten ist es, mit Hilfe der `debconf`-Funktion für den automatischen Start zu sorgen. Rufen Sie diese mit `dpkg-reconfigure ssh` auf, und beantworten Sie die Frage, ob Sie den `sshd`-Server starten wollen mit »Ja«.
SUSE	Hier müssen Sie dafür sorgen, dass SSH in dem Runlevel eingetragen ist, in dem Sie den Server laufen lassen wollen. Wenn Sie Linux nicht explizit in einem anderen Runlevel starten, sollten Sie mit dem YaST-Modul *runlevel* dafür sorgen, dass SSH im 2. Runlevel gestartet wird.

Tabelle 3.2: Unterschiede zwischen den Distributionen

3.2.2 Der erste Remote-Zugriff mit SSH

Auch wenn Sie jetzt noch keinen Client zur Hand haben, von dem aus Sie einen SSH-Zugriff auf Ihren Server wagen können, lässt sich der Zugriff über die interne Loopback-Schnittstelle simulieren.

SSH arbeitet mit privaten und öffentlichen kryptografischen Schlüsseln: Die öffentlichen Schlüssel werden dabei von den Clients genutzt, um die Daten zu verschlüsseln, die zum Server übertragen werden. Der Server entschlüsselt die Daten dann mit Hilfe seines privaten Schlüssels. Der erste Schritt zu einer sicheren Verbindung ist also der Austausch der Schlüssel. In der Praxis schaut der SSH-Client in einer in Ihrem Home-Vereichnis gespeicherten Datei nach, ob dort passend zum privaten Schlüssel des Servers bereits ein öffentlicher Schlüssel gespeichert ist. Beim ersten

Aufruf ist dies nicht der Fall; daher lädt der Client den öffentlichen Teil des Schlüssels herunter und speichert ihn dort ab. Da dies ein sicherheitskritischer Schritt ist, fragt der Client nach, ob wir diesem Schlüssel Vertrauen entgegenbringen[5]. Danach wird das Passwort abgefragt: Wenn Sie nicht explizit eine andere Nutzerkennung angeben, versucht der SSH-Client sich auf dem »entfernten« Rechner mit der gleichen Kennung anzumelden, mit der Sie auf dem System angemeldet sind, von dem aus Sie den Zugriff versuchen.

```
victor@hugo:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
RSA key fingerprint is 1c:9e:2c:91:59:50:7b:18:f9:05:5f:66:fd:27:b6:78.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost,127.0.0.1' (RSA) to the list of known hosts.
Password:# Passwort für Nutzer victor
# Die Begrüßungsmeldung des Servers
Last login: Fri Xxx 13 00:00:00 2005 from localhost
victor@hugo:~$ logout
victor@hugo:~$ cat .ssh/known_hosts
localhost ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAIEA3TBCZeggxiFoyXR5bm1cphChs6fcJEYviW//d7
qZjljqabnMOk8kL9lhDFLS2fsdO6urO2IFK1Vqmw5T12Ok7tvYeb9L2zQO4Ujn+puHUaqmMN9rsNg8o5d5Zj
KP9suYdHRzqi1SqAeA9jNlNzXbHA7xPOwAqj77mwF1pzGj/Ss=
victor@hugo:~$
```

Wenn die Autorisierung geklappt hat, erhalten Sie eine Shell, mit der Sie wie gewohnt arbeiten können. Außerdem wird beim ersten Aufruf in Ihrem Home-Verzeichnis ein Unterverzeichnis `.ssh` angelegt. Dort landen alle für die SSH-Nutzung mit dieser Kennung notwendigen Dateien, wie z. B. die Datei `known_hosts`, die die öffentlichen Schlüssel aller Server enthält, auf denen Sie sich bereits einmal eingewählt haben. Wir haben sie einmal mit `cat` ausgegeben. Die Einträge sind – wie hier jetzt nicht so genau zu erkennen – auf eine Zeile beschränkt: am Anfang steht der Name des Servers, dann der Typ des Schlüssels und danach der Schlüssel selbst. In dieser Datei können Sie auch »per Hand« Einträge ergänzen.

Sie können mit dem SSH-Client auch direkt ein Programm aufrufen:

```
victor@hugo:~$ ssh localhost date
Password:# Passwort für Nutzer victor
Fri Xxx 13 00:00:00 CET 2005
victor@hugo:~$
```

[5] Wir diskutieren an dieser Stelle nicht, welche Probleme hier auftreten könnten, werden aber später in Grundzügen darauf zurückkommen. Nur als Hinweis: eigentlich müsste der öffentliche Schlüsselteil auf einem anderen, sicheren Weg in die Datei `known_hosts` gelangen, da vor dem Austausch der Schlüssel die Verbindung noch nicht gegen Manipulationen geschützt ist. Theoretisch könnte jemand Ihnen mit einer »man-in-the-middle«-Attacke einen falschen Schlüssel unterschieben und danach die Verbindung zwischen Ihrem Client und dem Server belauschen.

In diesem Fall wird das Programm direkt ausgeführt und danach die Verbindung beendet. Da der öffentliche Schlüssel des Servers `localhost` jetzt bekannt ist, wird er nicht noch einmal abgefragt.

3.2.3 X-Forwarding mit SSH

Doch kommen wir zu den wirklich spannenden Dingen, mit denen wir Sie zum Staunen bringen wollen, vor allem, wenn Sie aus der Windows-Welt kommen. :-) Spätestens jetzt wäre es an der Zeit, einen zweiten Rechner mit einer Linux-Installation zur Hand zu haben, und sei es »nur« ein lauffähiges Knoppix. Wenn Sie Ihren Server mit `ping` finden, können wir Ihnen eine nützliche Funktion von SSH erklären, die eine nützliche Funktion von X noch nützlicher macht.

Zuerst müssen wir unseren SSH-Server aber noch dazu bewegen, das sogenannte X-Forwarding zuzulassen: damit ist gemeint, dass X-Programme, also Programme mit grafischer Ausgabe, ihr Fenster auf dem Client ausgeben können, während sie eigentlich auf dem Server ablaufen. Dazu muss jedoch auf der Client-Seite ein eigener X-Server laufen, weswegen wir diese Funktion am Beispiel einer zweiten Linux-Installation beschreiben wollen.

In der Konfigurationsdatei des `sshd` finden Sie gegen Ende einen Eintrag mit dem Namen `X11Forwarding`, der entweder auskommentiert oder auf »no« gesetzt ist. Setzen Sie diesen auf »yes«, und kontrollieren Sie, ob das `X11DisplayOffset` gesetzt ist.

Listing 3.4: /etc/ssh/sshd_config

```
...
X11Forwarding yes
X11DisplayOffset 10
...
```

Wenn Sie diese Änderungen gespeichert haben, müssen Sie den SSH-Server neu starten – wie immer am besten mit dem Init-Skript.

Wechseln wir nun die Konsole bzw. den Rechner. Im nächsten Beispiel wählen wir uns von einem mit Knoppix gebooteten Rechner in unseren Server ein:

```
knoppix@knoppix:~$ ssh -X victor@192.168.1.10
RSA key fingerprint is  1c:9e:2c:91:59:50:7b:18:f9:05:5f:66:fd:27:b6:78.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.10' (RSA) to the list of known hosts.
Password:# Passwort für Nutzer victor
# Die Begrüßungsmeldung des Servers
Last login: Fri Xxx 13 00:00:00 2005 from localhost
knoppix@knoppix:~$ xclock&
knoppix@knoppix:~$
```

Die Daten werden für die X-Clients transparent verschlüsselt übertragen. Versuchen Sie es einmal, und starten Sie z. B. `xclock`, ein einfaches X-Programm, das aber zu den Basispaketen von X gehört und auf jedem Rechner zu finden sein sollte.

Hinweis: Wenn's mal nicht so geklappt hat ...

Problem: Es kann keine Verbindung aufgebaut werden, und Sie erhalten als Fehlermeldung »`ssh: connect to host hugo port 22: Connection timed out`«. Oder der Server schließt die Verbindung direkt wieder: »`ssh_exchange_identification: Connection closed by remote host`«; oder die Verbindung wird zurückgewiesen: »`ssh: connect to host hugo port 22: Connection refused`«

Analyse: Das Gegenüber kann eine Verbindung im Prinzip auf zwei Ebenen ablehnen: Die erste Möglichkeit ist, dass eine Firewall-Einstellung dafür sorgt, dass die Pakete den SSH-Server gar nicht erst erreichen. In diesem Fall erhalten Sie das erste Symptom: Es werden nur Pakete an den Server gesendet, aber es kommen keine zurück, daher »Connection timed out«.

Die zweite Möglichkeit, bei denen Ihnen zumindest gesagt wird, dass Ihnen der Server die Türe vor der Nase zugeschlagen hat, deutet darauf hin, dass der SSH-Server bewußt Ihre Anfrage ablehnt. Ein Blick in die passende Log-Datei sollte dieses Verhalten belegen:

```
hugo:~# tail /var/log/auth.log
Xxx 13 00:00:00 hugo sshd[5880]: refused connect from 192.168.1.11
                                                        (192.168.1.11)
hugo:~#
```

In diesem Fall wurde dem SSH-Server mitgeteilt, dass entweder die Kennung oder der Rechner keine Erlaubnis hat, sich anzumelden. Diese Informationen können in der Konfigurationsdatei `sshd_config` oder über den `hosts_access`-Mechanismus, also Einträge in den Dateien `/etc/hosts.deny` und `/etc/hosts.allow` eingetragen worden sein.

»Connection refused« erhalten Sie in der Regel, wenn auf dem Port keine Anfragen beantwortet werden. Das kann bedeuten, dass der SSH-Server gar nicht läuft oder auf einer anderen Schnittstelle auf Anfragen wartet.

Lösung: Bei einem **Timeout** sollten Sie auf dem Zielserver für einen SSH-Versuch die Firewall deaktivieren bzw. direkt in der Firewall-Konfiguration dafür sorgen, dass SSH-Zugriffe erlaubt sind. In den anderen Fällen können Sie die Datei `/etc/hosts.allow` probeweise um eine Zeile `sshd: ALL` ergänzen. Sollte es danach funktionieren, müssten Sie nur noch das »ALL« durch auf Ihren Anwendungsfall passende IP-Bereiche ersetzen. Ansonsten hilft nur noch ein Blick in die Kon-

figurationsdatei des SSH-Dämons. Aber zuerst sollten Sie kontrollieren, ob er überhaupt läuft. ;-)

Problem: SSH beschwert sich beim Start der Verbindung mit der Meldung:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED        @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
04:07:77:c0:5a:6e:70:c2:1f:3e:4b:56:a2:b8:b3:38.
Please contact your system administrator.
Add correct host key in /home/victor/.ssh/known_hosts to get rid of this
message.
Offending key in /home/victor/.ssh/known_hosts:1
RSA host key for hugo has changed and you have requested strict checking.
Host key verification failed.
```

Analyse: Dies bedeutet, dass sich entweder das Schlüsselpaar geändert hat, mit dem der SSH-Server arbeitet oder dass unter seiner IP-Nummer in der Datei `.ssh/known_hosts` ein Schlüssel eines anderen Rechners aufgeführt ist. Solche Probleme treten häufig in Situationen auf, in denen Sie sich auf Rechnern einloggen, die per DHCP dynamisch eine IP-Nummer zugewiesen bekommen. Hier kann es passieren, dass ein Rechner eine IP-Nummer zugewiesen bekommt, die vorher einem anderen Rechner gehörte. Dieses Problem ist aber nicht von einer »man-in-the-middle attack« zu unterscheiden, daher die Warnung.

Lösung: Wenn Sie sich sicher sind, dass sich hier niemand zwischen Ihren Client und den Server schalten möchte, auf den Sie zugreifen wollen, reicht es aus, die jeweilige Zeile in der Datei `.ssh/known_hosts` zu löschen. In der Meldung ist übrigens ein Hinweis auf die problematische Zeile: `Offending key in /home/victor/.ssh/known_hosts:1`, also in der 1. Zeile der Datei `known_hosts`. Nach dem Löschen des Eintrags wird Ihnen beim nächsten Zugriff der Schlüssel des Servers erneut zum Download angeboten, und Sie haben wieder eine Verbindung.

3.3 Mit ganzen Desktops durch das Netz

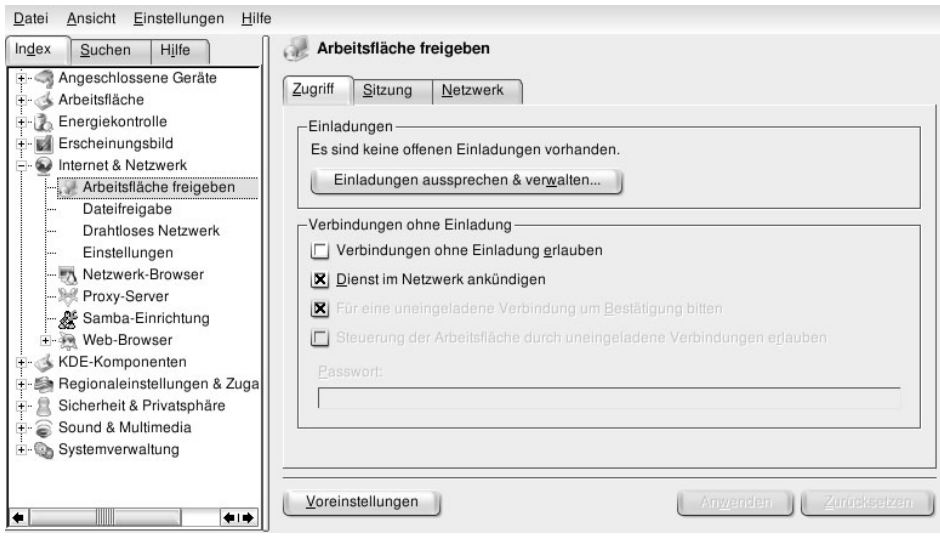
Auch wenn wir das Verschicken einzelner Fenster (und Programme) über das Netzwerk der Übertragung eines kompletten Desktops jederzeit vorziehen würden, seien hier doch die zurzeit wichtigsten Technologien erwähnt, die für den Remote-

Zugriff auf Desktops existieren. Mit der Software, die wir Ihnen jetzt vorstellen wollen, können Sie auch auf Windows-Rechner zugreifen. Falls Sie den Einsatz von Linux auf Arbeitsplatz-Rechnern planen, können Ihnen diese Tools helfen, Software zur Verfügung zu stellen, die z. B. nur für Windows oder ein anderes proprietäres Betriebssystem existiert.

Das Virtual Network Computing (VNC) ist wohl das am weitesten verbreitete Protokoll für einen Remote-Zugriff auf eine grafische Oberfläche, das sowohl als Client- als auch als Server-Lösung auf nahezu allen Plattformen implementiert ist. VNC hat jedoch den Nachteil, dass auf dem Zielrechner ein laufender Desktop existieren muss, der dann über das Netzwerk übertragen werden kann. Das bedeutet i. d. R., dass auf diesem Rechner ein Nutzer angemeldet sein und den Zugriff freigegeben haben muss. Damit eignet sich VNC z. B. gut für den Support, um Nutzern per Telefon und Remote-Zugriff bei Problemen zu helfen.

Wir wollen Ihnen am Beispiel von KDE zeigen, wie einfach die Freigabe eines Desktops und der Zugriff darauf ist. Dazu brauchen wir zwei Linux-Rechner, an denen sich jeweils ein Nutzer mit der grafischen Oberfläche KDE angemeldet hat. In unserem Szenario hat sich ein Kollege per Telefon gemeldet und benötigt Unterstützung bei einem Linux-Problem.

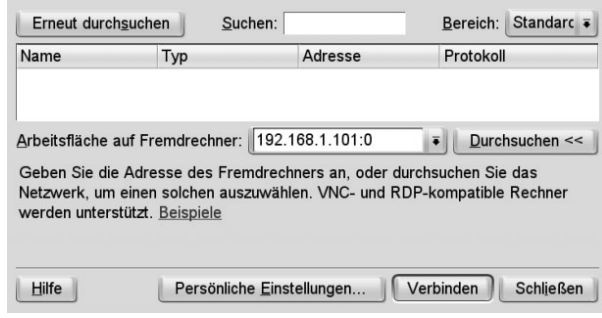
Zuerst muss der Kollege die Arbeitsfläche freigeben, damit wir von unserem Rechner aus darauf zugreifen können. Dies geht am einfachsten über das Kontrollzentrum von KDE. Dort findet er unter der Rubrik INTERNET & NETZWERK den Punkt ARBEITSFLÄCHE FREIGEBEN.



Dort wählt er den Button EINLADUNG AUSSPRECHEN & VERWALTEN und ruft im folgenden Dialog den Punkt NEUE PERSÖNLICHE EINLADUNG auf. Jetzt wird ein Server-Prozess gestartet, der auf einen Kontakt wartet. Zusätzlich ist ein Passwort

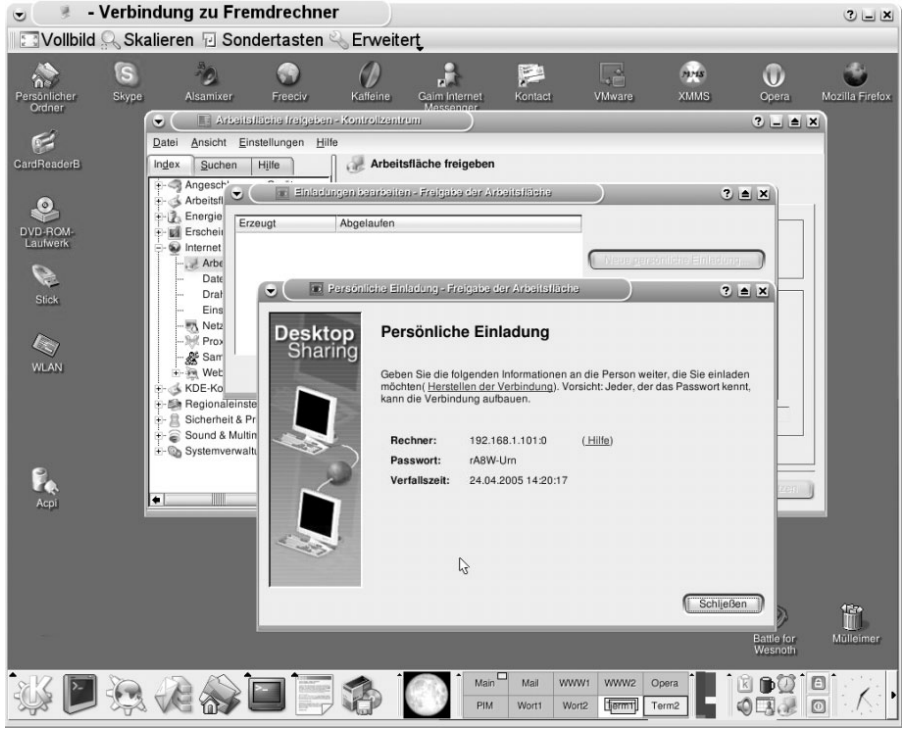
generiert worden, mit dem wir uns autorisieren müssen, um den Zugriff auf den Rechner zu erhalten.

Wir gehen einmal davon aus, dass der Kollege diese Aktionen mit unserer Hilfe über das Telefon hat ausführen können. Wir starten jetzt aus dem K-Menü unter der Rubrik INTERNET die VERBINDUNG ZU FREMDRECHNER; dahinter verbirgt sich die Software Krdc. Ein Versuch wäre, das Netzwerk nach der Freigabe zu durchsuchen. Wenn hier jedoch keine Rechner angezeigt werden, findet der Kollege am Telefon in seinem Freigabe-Dialog in der Zeile RECHNER die Informationen, die wir in das Feld FREMDRECHNER eintragen müssen.



Der nächste Dialog ermöglicht es uns, die Qualität der Verbindung festzulegen. Dies hat Auswirkungen darauf, welche Informationen wie über das Netzwerk übertragen werden, um ein flüssiges Arbeiten zu ermöglichen. Hier tun Sie sich keinen Gefallen, wenn Sie die Bandbreite überschätzen; bewegt sich die Maus später nur stockend und ruckend, sollten Sie die Verbindung noch einmal mit einer niedrigeren, angenommenen Bandbreite aufbauen. Jetzt wird der Kontakt hergestellt. Das Gegenüber muss jetzt den Zugriff durch uns bestätigen.

Danach werden wir nach dem Passwort gefragt, das für die Verbindung angelegt worden ist. Der Kollege am Telefon sollte bei der Mitteilung nicht vergessen, auf Groß- und Kleinschreibung hinzuweisen, da diese für die Anmeldung wichtig ist. Dann merken wir durch einem kleinen Fortschrittsdialog, dass versucht wird, die Verbindung herzustellen. Nach kurzer Zeit sollte dann ein Fenster erscheinen, in dem der Desktop des Kollegen angezeigt wird.



Innerhalb dieses Fensters wirken unsere Mausaktionen so, als würden wir vor diesem Rechner sitzen. Im Übrigen kann der Kollege alles mitverfolgen, was wir jetzt tun.

Für den Zugriff auf einen bildschirmlosen Server existiert mit TightVNC eine Software, die sich z. B. über eine SSH-Verbindung auf einem entfernten Rechner starten lässt. Danach kann dann über die Option VERBINDUNG ZU FREMDRECHNER eine VNC-Verbindung aufgebaut werden.

```
victor@hugo:~$ tightvncserver --help
TightVNC server version 1.2.9

Usage: tightvncserver [<OPTIONS>] [:<DISPLAY#>]
       tightvncserver -kill :<DISPLAY#>

<OPTIONS> are Xtightvnc options, or:

        -name <DESKTOP-NAME>
        -depth <DEPTH>
        -geometry <WIDTH>x<HEIGHT>
        -httpport number
        -basehttpport number
        -alwaysshared
        -nevershared
        -pixelformat rgb<NNN>
        -pixelformat bgr<NNN>

See tightvncserver and Xtightvnc manual pages for more information.
victor@hugo:~$ tightvncserver :5

You will require a password to access your desktops.

Password:
Verify:

New 'X' desktop is 192.168.1.10:5

Creating default startup script /home/victor/.vnc/xstartup
Starting applications specified in /home/victor/.vnc/xstartup
Log file is /home/victor/.vnc/hugo:5.log

victor@hugo:~$ tightvncserver -kill :5
Killing Xtightvnc process ID 19730
victor@hugo:~$ tightvncserver :5

New 'X' desktop is 192.168.1.10:5
```

```
Starting applications specified in /home/victor/.vnc/xstartup
Log file is /home/victor/.vnc/hugo:5.log

victor@hugo:~$
```

Der Start eines VNC-Servers ist dabei recht einfach. In der Regel genügt der Aufruf des Wrapper-Scripts `tightvncserver`. Alle wichtigen Angaben werden dann abgefragt und der Server unter der nächsten freien Display-Nummer gestartet. Es kann aber auch, wie wir es im obigen Beispiel getan haben, eine spezifische Display-Nummer angegeben werden. Zusätzlich wird im Konfigurationsverzeichnis unter anderem ein Script angelegt, das einen Window-Manager startet. Hier können Sie z. B. eintragen, das KDE oder vielleicht auch ein etwas ressourcensparenderer Desktop aufgebaut wird.

Der zweite Aufruf geht dann übrigens schneller, da TightVNC auf die beim ersten Aufruf gemachte Konfiguration zurückgreift. Sie beenden den VNC-Server mit dem Aufruf `tightvncserver -kill DISPLAYNR`; DISPLAYNR ergänzen Sie durch die Display-Nummer, mit der der Server gestartet wurde. Sie können auch mehrere VNC-Server unter verschiedenen Display-Nummern starten, falls dies notwendig sein sollte.

Den TightVNC-Server gibt es auch für Windows, jedoch bietet sich hier die Nutzung des RDP-Protokolls an, das die neueren Windows-Versionen von Haus aus mitbringen. Krdc bietet zusätzlich auch die Möglichkeit, eine RDP-Verbindung aufzubauen. Nur muss dazu i. d. R. noch rdesktop installiert werden; diese Software übernimmt die eigentliche Aufgabe des RDP-Zugriffs.

3.4 Auf der Höhe der Zeit mit (x)ntp

Ein Aspekt, den man oft vernachlässigt, dem aber bei der Administration einer wahrscheinlich immer größer werdenden Gruppe von Netzwerkrechnern eine wichtige Bedeutung zukommt, ist die Synchronisation der Systemzeit auf den verschiedenen Rechnern. Diese Aufgabe nehmen uns das **N**etwork **T**ime **P**rotocol und die nach ihm benannte Software (x)ntp (Port 123) ab.

Die Software bietet zwei Tools, die dazu dienen, die Systemzeit mit einem Zeitserver synchron zu halten: `ntpdate` setzt die Systemzeit, während der `ntpd` die Systemzeit aktuell hält. Was ist wann nötig? Wenn sich nicht gerade eine Funkuhr auf dem Motherboard des Rechners befindet, dessen Systemzeit aktuell gehalten werden muss, dann muss man leider davon ausgehen, dass die interne Rechneruhr mit der Zeit mehr oder weniger stark aus dem Ruder läuft. Dies geschieht aber in der Regel in Form berechenbarer Abweichungsintervalle. Der `ntpd` ermittelt durch die über einen längeren Zeitraum ablaufenden Synchronisationsprozesse Werte für diese Abweichung und kann dadurch die Systemzeit so genau halten, wie sie der Zeitserver liefert. Die Änderungen, die der NTP-Dämon an der Systemzeit vornimmt, liegen dabei in für das Funktionieren des Systems unkritischen Bereichen.

Wenn Sie die Möglichkeiten des NTP nutzen wollen, müssen Sie nicht unbedingt selbst einen Server betreiben, der an eine Atomuhr angeschlossen ist. Es gibt eine Reihe von Servern im Netzwerk, die als Zeitserver fungieren können. Das Projekt NTP, das die Software programmiert hat, führt eine Liste der öffentlichen NTP-Server. Außerdem stellt es mit den `pool`-Adressen einen Service zur Verfügung, der eine gleichmäßige Auslastung der öffentlichen Server gewährleistet. Wenn Sie die Adresse `de.pool.ntp.org` benutzen, wird Ihnen automatisch ein Zeitserver zugewiesen. Wenn Sie diese Adresse z. B. mehrfach in der Konfigurationsdatei für den `ntpd` eintragen, wird dieser sich mit verschiedenen Servern synchronisieren. Damit ist gewährleistet, dass für diejenigen, die diesen Service kostenfrei zur Verfügung stellen, kein unnötiger Netzwerkverkehr entsteht.

Beginnen wir jedoch von vorn: zuerst sollten Sie die Uhrzeit des Rechners auf einen möglichst aktuellen Wert stellen. Wenn der Rechner bereits ans Internet angeschlossen ist, können Sie diese Aufgabe dem Programm `ntpdate` überlassen.

Achtung

Wir möchten an dieser Stelle zur Vorsicht raten! Achten Sie darauf, dass die Uhrzeit des Rechners, auf dem Sie die Systemzeit aktualisieren wollen, bereits möglichst nahe an der realen Zeit liegt. Bei einem größeren Zeitsprung könnten die laufenden Prozesse sonst auf »dumme Gedanken« kommen! Ein plötzlich anspringender Bildschirmschoner ist dabei noch das geringste Übel …

Ein Blick auf `date` gibt uns die aktuelle Systemzeit aus. Diese liegt – wie ein Vergleich mit der Armbanduhr oder sonst einem erreichbaren Chronometer zeigt – im Bereich der aktuellen Zeitrechnung. Jetzt können wir mit `ntpdate` die Zeit genauer stellen lassen:

```
hugo:~# date
Mon xxx 16 09:06:41 CEST 2005
hugo:~# ntpdate de.pool.ntp.org
16 xxx 09:07:17 ntpdate[6487]: step time server 84.16.227.162 offset 26.936620 sec
hugo:~# ntpdate de.pool.ntp.org
16 xxx 09:07:22 ntpdate[6490]: adjust time server 84.16.227.162 offset -0.000293 sec
hugo:~#
```

Sie erkennen beim zweiten Aufruf, dass der Offset zum Zeitserver nur noch im Bereich von zehntausendstel Sekunden liegt. Mit dieser Genauigkeit lässt sich jetzt der Zeitserver starten:

```
hugo:~# /etc/init.d/ntp-server start
hugo:~# ntpq -p
```

```
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 mabuse.homeip.n 192.53.103.104   2 u    5   64    1   39.181   10.142   0.004
 binky.tuxfriend 192.53.103.104   2 u    4   64    1   29.935   x8.660   0.004
 anton.hin.de    130.149.17.21    2 u    3   64    1   24.276   11.959   0.004
 LOCAL(0)        LOCAL(0)        13 l    2   64    1    0.000    0.000   0.004
hugo:~#
```

Wir haben in der Konfigurationsdatei dreimal einen Eintrag `de.pool.ntp.org` gemacht: Während `ntpdate` zur besseren Ermittlung des Offsets den gleichen Server zugewiesen bekommt, erhält der `ntpd` drei verschiedene Server, mit denen er sich synchronisieren kann.

NTP für das lokale Netzwerk konfigurieren: Wir haben die Konfigurationsdatei jetzt schon zweimal erwähnt, nun wollen wir Ihnen die wichtigen Einträge kurz vorstellen: Wenn Sie nur einen Client einrichten, müssen Sie in der Konfigurationsdatei nur einen oder mehrere Server angeben, mit denen die Zeit synchronisiert werden soll. Hier sehen Sie die Einträge, die wir für das obige Beispiel genutzt haben.

Listing 3.5: /etc/ntp.conf

```
...
# pool.ntp.org maps to more than 100 low-stratum NTP servers.
# Your server will pick a different set every time it starts up.
#  *** Please consider joining the pool! ***
#  ***  <http://www.pool.ntp.org/#join>  ***
#server pool.ntp.org
server de.pool.ntp.org
server de.pool.ntp.org
server de.pool.ntp.org
```

Wenn Sie im lokalen Netzwerk einen Zeitservice verwenden wollen, sollten Sie jedoch nicht alle Rechner mit externen Servern synchronisieren. Es reicht, zwei oder drei Rechner dafür einzurichten. Diese können dann als lokale Zeitserver fungieren. Damit diese aber auch als Server von den Clients akzeptiert werden, sollten Sie ihnen einen höheren `stratum`-Wert geben, als in der Konfigurationsdatei als Standard eingetragen ist.

Listing 3.6: /etc/ntp.conf

```
# ... and use the local system clock as a reference if all else fails
# NOTE: in a local network, set the local stratum of *one* stable server
# to 10; otherwise your clocks will drift apart if you lose connectivity.
server 127.127.1.0
fudge 127.127.1.0 stratum 10
```

Das Stratum dient dem Netzwerk der Zeitserver als Identifikation der Wichtigkeit bzw. Nähe eines Servers zu einer Zeitquelle, z. B. einer Funkuhr. Die Rechner, die

direkt an eine solche Quelle angeschlossen sind, erhalten das Stratum 0; diejenigen, die diese zum synchronisieren nutzen, das Stratum 2 usw. Ab einem gewissen Stratum-Wert nehmen die NTP-Dämonen die Zeitinformationen eines Servers nicht mehr »ernst« und verlassen sich lieber auf die eigene Hardware-Uhr, auch wenn die Zeit des genutzten Servers genauer ist.

Automatische Uhren im lokalen Netz: Sie können sich die Konfiguration der NTP-Clients vereinfachen, wenn Sie die Zeitserver ihre Uhrzeit per Broadcast propagieren lassen. Dazu ergänzen Sie auf den Servern in der Konfiguration eine `restrict`- und eine `broadcast`-Zeile:

Listing 3.7: /etc/ntp.conf

```
...
# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1 nomodify
restrict 192.168.1.0 mask 255.255.255.0 nomodify

# If you want to provide time to your local subnet, change the next line.
broadcast 192.168.1.255
...
```

Die Clients müssen jetzt noch darauf geeicht werden, sich die Informationen über die Zeit per Broadcast zu suchen. Dazu kommentieren Sie die letzten beiden Zeilen in der Konfigurationsdatei aus:

Listing 3.8: /etc/ntp.conf

```
# If you want to listen to time broadcasts on your local subnet,
# de-comment the next lines. Please do this only if you trust everybody
# on the network!
disable auth
broadcastclient
```

Das von uns hier beschriebene Beispiel arbeitet ohne Authentisierung. Nehmen Sie die Warnung in der Konfigurationsdatei daher ernst. Informationen zu Authentisierung und weitere wichtige Informationen finden Sie in den unten angegebenen Literaturhinweisen.

Zum Schluss noch ein Beispiel für eine etwas langsamere Synchronisation der Systemzeit: der von uns installierte Rechner ging ca. 15 Minuten nach. Anstatt die Systemzeit mit `ntpdate` zu setzen, haben wir den NTP-Dämon gestartet und eine Weile laufen lassen.

Listing 3.9: /var/log/daemon.log

```
Xxx 29 18:25:15 theodor ntpd[6243]: ntpd 4.2.0a@1:4.2.0a+stable-8-r Sat Mar 19 14:14:09
                                                                              CET 2005 (1)
Xxx 29 18:25:15 theodor ntpd[6243]: precision = 3.000 usec
```

```
Xxx 29 18:25:15 theodor ntpd[6243]: Listening on interface wildcard, 0.0.0.0#123
Xxx 29 18:25:15 theodor ntpd[6243]: Listening on interface wildcard, ::#123
Xxx 29 18:25:15 theodor ntpd[6243]: Listening on interface lo, 127.0.0.1#123
Xxx 29 18:25:15 theodor ntpd[6243]: Listening on interface eth0, 192.168.1.10#123
Xxx 29 18:25:15 theodor ntpd[6243]: kernel time sync status 0040
Xxx 29 18:28:32 theodor ntpd[6243]: synchronized to LOCAL(0), stratum 13
Xxx 29 18:29:34 theodor ntpd[6243]: synchronized to 193.218.127.251, stratum 2
Xxx 29 18:59:55 theodor ntpd[6243]: time reset +856.891493 s
```

Der `ntpd` hat in diesem Beispiel eine ganze Weile gewartet und dann die Uhrzeit so vorgestellt, dass bei der Angleichung keine Stundengrenze überschritten wurde.

Weitere Literatur

- `www.ntp.org`
- »TimePrecision-HOWTO«
- `man ntpd`
- `man ntpdate`

3.5 Einrichtung eines einfachen DNS-Proxys/Servers

Jetzt haben Sie den ersten Server und vielleicht auch schon den ersten Client eingerichtet; da liegt es nahe, den Kindern Namen zu geben und die Rechner auch darüber anzusprechen. Dazu benötigt man den sogenannten Nameservice bzw. Domain Name ServiceDomain Name System, kurz DNS (siehe auch Abschnitt 2.1.2 auf S. 23). Dieser läuft als Server-Prozess auf einem Rechner im Netzwerk (häufig dem Gateway), mit dessen Hilfe die Clients die Namen in IP-Nummern auflösen können. Standard für diesen Dienst ist der BIND-Server, der **B**erkley **I**nternet **N**ame **D**omain Server, der sich rühmen kann, die meistgenutzte DNS-Server-Software im Internet zu sein. Andererseits ist er aber auch ein Monstrum und die Pflege eines DNS-Servers ist für einen Anfänger eine Strafe. Daher wollen wir Ihnen für Ihr lokales Netzwerk eine andere Lösung anbieten, die Ihnen helfen wird, einen internen Domain Name Service aufzusetzen. Wenn Sie jedoch eigene Domains selbst verwalten wollen, kommen Sie um BIND nicht herum.

Was tut DNSMasq für Sie?

- Sie können in der Datei `/etc/hosts` auf einfache Weise Namen und IP-Nummern für die Rechner in Ihrem lokalen Netzwerk vorhalten. DNSMasq liest diese Datei und liefert sie auf Anfrage aus.
- DNSMasq kann als Proxy für Ihr lokales Netzwerk arbeiten. Das heißt, dass alle DNS-Anfragen über einen Server abgewickelt werden können, der diese zwischenspeichert. Dadurch wird die Außenverbindung entlastet, und die Namensauflösung wird deutlich beschleunigt.
- Zusätzlich bietet DNSMasq noch einfache DHCP-Funktionen an, die wir in Kapitel 5 ab S. 89 jedoch ausführlicher anhand der DHCP-Software des Internet System Consortium erläutern.

dnsmasq	Debian	Fedora	SUSE
dnsmasq	dnsmasq	dnsmasq	dnsmasq
nslookup Tools zum Test des DNS-Systems	dnsutils	bind-utils	bind-utils

Tabelle 3.3: Paketübersicht

Nach der Installation finden Sie im Verzeichnis `/etc` die Konfigurationsdatei `dnsmasq.conf`. Diese muss je nach Anwendungsfall noch ein wenig angepasst werden. Betrachten wir dazu zwei Fälle:

Fall 1: Server mit statischer IP-Nummer

Im einfachen Fall wollen Sie einen Ihrer Rechner im lokalen Netzwerk, bei dem die Netzwerkanbindung bereits funktioniert, um die DNS-Proxy-Funktion erweitern. In diesem Fall brauchen Sie an der Standard-Konfiguration eigentlich nichts zu ändern! Wie arbeitet DNSMasq in diesem Fall?

- Die Informationen, welchen DNS-Server DNSMasq nach Adressen fragen soll, werden der Datei `/etc/resolv.conf` entnommen.
- DNSMasq stellt den Service automatisch auf jeder Netzwerkschnittstelle des Rechners zur Verfügung. Die Anfragen werden auf dem Port 53 beantwortet, dem Standard für den Domain Name Service.

Wenn Sie also DNSMasq nach der Installation über das Init-Script starten, sollten eine einfache Anfrage bereits funktionieren:

```
hugo:~# /etc/init.d/dnsmasq start
Starting DNS forwarder and DHCP server: dnsmasq.
hugo:~# nslookup www.uni-duesseldorf.de localhost
Server:         localhost
Address:        127.0.0.1#53

Name:   www.uni-duesseldorf.de
Address: 134.99.128.40
hugo:~#
```

Zum Testen nutzen wir das Tool `nslookup`, das i. d. R. auf Ihrem System bereits installiert ist. Sie können es mit einer oder mit zwei Optionen aufrufen: Wenn Sie testen möchten, ob die lokale DNS-Konfiguration funktioniert, rufen Sie `nslookup` nur mit einem Rechnernamen auf, der aufgelöst werden soll; wenn Sie die Funktion eines im Netz befindlichen DNS-Servers testen wollen, geben Sie `nslookup` noch die Adresse dieses Servers mit, damit es dort versucht, den Namen aufzulösen.

Jetzt wollen wir einen »privaten« Rechnernamen definieren, den wir im lokalen Netzwerk nutzen wollen. Dazu ergänzen wir die Datei `/etc/hosts` um eine Zeile für unseren Client:

Listing 3.10: /etc/hosts

```
192.168.1.11  theodor
192.168.1.12  antonius
```

Nach einem Neustart des Server-Prozesses liefert uns DNSMasq auch die IP-Nummern dieses Rechners auf Anfrage zurück. Dass dies vom Rechner `hugo` aus funktioniert, glauben Sie uns wahrscheinlich. Dazu wäre DNSMasq im Prinzip auch nicht notwendig gewesen, da von den dort laufenden Programmen die Datei `/etc/hosts` direkt ausgewertet wird. Loggen Sie sich daher einmal auf einem Ihrer Linux-Clients ein und versuchen Sie von dort aus, eine DNS-Anfrage beantwortet zu bekommen:

```
victor@theodor:~$ nslookup antonius 192.168.1.10
Server:         192.168.1.10
Address:        192.168.1.10#53

Name:   antonius
Address: 192.168.1.12
victor@theodor:~$
```

Dieses Ergebnis ist schon interessanter: Wir haben über die IP-Nummer unseres Servers nach einem Hostnamen gefragt und eine Antwort bekommen. Jetzt könnten wir die Datei `/etc/resolv.conf` dahingehend ändern, dass wir dort diesen Server als DNS-Server eintragen:

Listing 3.11: /etc/resolv.conf

```
nameserver 192.168.1.10
```

Ab jetzt werden von diesem Client aus alle Anfragen über unseren DNS-Proxy abgewickelt, inklusive der dort in der Datei `/etc/hosts` gespeicherten Namen. Dadurch können Sie jetzt von theodor auf antonius über dessen Namen zugreifen.

Jetzt können Sie auf allen Clients die 192.168.1.10 als DNS-Server eintragen, nur auf dem Server selbst wird noch direkt auf die »Originale« zugegriffen. Dies liegt an den Einträgen in der `/etc/resolv.conf`. Würden wir hier ohne zusätzliche Maßnahmen die 127.0.0.1 eintragen, würden alle lokalen Programme sich an DNSMasq wenden, aber DNSMasq selber leider auch. Glücklicherweise existiert eine Möglichkeit, dieses Henne-Ei-Problem zu umgehen: Sie können die Server, an die DNSMasq die Anfragen weiterleiten soll, in der Konfigurationsdatei angeben:

Listing 3.12: /etc/dnsmasq.conf

```
[...]
# If you don't want dnsmasq to read /etc/resolv.conf or any other
# file, getting its servers from this file instead (see below), then
# uncomment this
```

```
no-resolv

# If you don't want dnsmasq to poll /etc/resolv.conf or other resolv
# files for changes and re-read them then uncomment this.
no-poll

# Add other name servers here, with domain specs if they are for
# non-public domains.
server=nnn.nnn.nnn.nnn # Ein erster Nameserver
server=nnn.nnn.nnn.nnn # Ein zweiter Nameserver
[...]
```

Nach einem Neustart des DNSMasq-Servers können Sie jetzt die Datei `/etc/resolv.conf` auf eine Zeile der Art `nameserver 127.0.0.1` reduzieren. Dadurch werden alle lokalen Anfragen auch über DNSMasq abgewickelt.

Fall 2: Server als Gateway für die Einwahl ins Internet

Wenn Sie sich mit Ihrem Rechner über Modem, ISDN oder DSL ins Internet einwählen, ist der Einsatz von DNSMasq besonders wünschenswert, da die Netzwerkverbindung ja nicht auch noch durch häufige DNS-Anfragen belastet werden muss. Hier haben wir aber i. d. R. das Problem, dass bei der Einwahl die IP-Nummer dynamisch vergeben wird. Was ist zu tun?

- Nach »Außen« bei der Weiterleitung der Anfragen ergeben sich Probleme, weil wir DNSMasq bei jeder Einwahl die aktuell gültigen DNS-Server mitteilen müssen, insbesondere dann, wenn der Gateway-Rechner selbst auch über DNSMasq DNS-Anfragen absetzen soll.
- Nach »Innen« bleibt alles beim Alten; es sollte nur dafür gesorgt werden, dass DNSMasq nicht auf der Einwahl-Schnittstelle DNS-Anfragen beantwortet.

Warum gibt es bei der Einwahl Probleme? Da jeder Provider i. d. R. eigene Nameserver besitzt und damit sich je nach Einwahl auch deren IP-Nummern ändern, verändert der für die Einwahl zuständige Dämon die Datei `/etc/resolv.conf`, um sie den aktuellen Gegebenheiten anzupassen. Das würde aber unseren Eintrag `nameserver 127.0.0.1` überschreiben. Also müssen wir DNSMasq auf andere Weise mitteilen, wo die aktuell gültigen Informationen bezüglich der zu nutzenden DNS-Server stehen.

Der PPP-Dämon, der für die Einwahl über Modem und DSL genutzt wird (und der `ipppd` für die Einwahl per ISDN auch) kann so konfiguriert werden, dass er eine separate `resolv.conf` anlegt und darin die aktuellen DNS-Server speichert. Dies wird mit Hilfe der Option `usepeerdns` angeschaltet, die Sie in der Datei `/etc/ppp/options` eintragen können. Danach müssen Sie nur noch DNSMasq mitteilen, wo die auszuwertende `resolv.conf` zu finden ist:

Listing 3.13: /etc/dnsmasq.conf

```
[...]
# Change this line if you want dns to get its upstream servers from
# somewhere other that /etc/resolv.conf
resolv-file=/etc/ppp/resolv.conf

# By  default,  dnsmasq  will  send queries to any of the upstream
# servers it knows about and tries to favour servers to are  known
# to  be  up.  Uncommenting this forces dnsmasq to try each query
# with  each  server  strictly  in  the  order  they   appear   in
# /etc/resolv.conf
#strict-order

# If you don't want dnsmasq to read /etc/resolv.conf or any other
# file, getting its servers from this file instead (see below), then
# uncomment this
#no-resolv

# If you don't want dnsmasq to poll /etc/resolv.conf or other resolv
# files for changes and re-read them then uncomment this.
#no-poll
[...]
```

Sie sollten darauf achten, dass die Optionen `no-resolv` und `no-poll` in diesem Fall *nicht* aktiviert sind!

Debian-Tipp

Die Debian-Entwickler haben diese Probleme mit einem eigenen Tool für die Pflege der `/etc/resolv.conf` umgangen. Wenn Sie DNSMasq zusammen mit dem Paket `resolvconf` installieren, werden alle oben beschriebenen Schritte automatisch erledigt. Außerdem ist die Software PPP-Dämon und alle weiteren, die die `/etc/resolv.conf` manipulieren, ebenfalls an das Paket `resolvconf` angepasst worden. Das geht soweit, dass bei einer Änderung der DNS-Konfiguration alle Prozesse, die diese Informationen nutzen, darüber informiert respektive neu gestartet werden, wenn das notwendig ist.

Um DNSMasq dazu zu überreden, die Einwahl-Schnittstelle nicht zu bedienen, haben Sie zwei Möglichkeiten: Sie können entweder die Schnittstelle ausschließen, oder explizit die IP-Nummern angeben, auf denen DNSMasq Anfragen beantworten soll:

- Im ersten Fall sorgen Sie mit der Option `except-interface=ppp0` z. B. dafür, dass DNSMasq das Interface `ppp0` ignoriert, das bei der Einwahl als ausgehende Netzwerkschnittstelle angelegt wird.

- Im zweiten Fall geben Sie mit der Option `listen-address=192.168.1.10` z. B. an, dass DNSMasq nur an der Schnittstelle antworten soll, die mit dieser IP-Nummer konfiguriert ist. Vergessen Sie in diesem Fall nicht, eine weitere Zeile `listen-address=127.0.0.1` hinzuzufügen, damit auch lokale Anfragen verarbeitet werden!

Je nach Einsatzszenario kann das eine oder andere Verfahren sinnvoller sein. Diese Entscheidung wollen wir aber Ihnen überlassen. Weitere Hinweise zu DNSMasq finden Sie im Dokumentationsverzeichnis, das mit dem Paket installiert wird oder auf der Webseite des Projekts.

Weitere Literatur

- Homepage des Projekts: `http://thekelleys.org.uk/dnsmasq/doc.html`
- `/usr/share/doc/dnsmasq/setup.html`

4 Drucken mit CUPS

Unter Linux einen Drucker ans Laufen zu bekommen gehört sicher nicht unbedingt in ein Buch über das Netzwerken mit Linux. Jedoch müssen für das Drucken über das Netzwerk einige Voraussetzungen erfüllt sein, die es notwendig machen, hier kurz auf die Installation des mittlerweile zum Standard gewordenen Drucksystems CUPS (**C**ommon **U**nix **P**rinting **S**ystem) einzugehen.

CUPS (Common Unix Printing System)	Debian	Fedora	SUSE
Standard-CUPS-Server	cupsys	cups	cups
Druck-Clients Clients, die für das Drucken mit CUPS benötigt werden	cupsys-client cupsys-bsd	cups	cups-client
Bibliotheken und Toolprogramme Toolprogramme für die Konvertierung von Postscript-Dateien in das Druckformat	gs-esp hpijs hplip hpoj cupsys-driver-gimpprint	ghostscript hpijs gimp-print-cups	ghostscript-serv hplip-hpijs cups-driver-stp
Druckertreiber Druckerkonfigurationen und Tools, die eine solche erstellen	foomatic-db foomatic-db-engine foomatic-db-hpijs foomatic-db-gimp-print foomatic-filters foomatic-filters-ppds	foomatic	cups-SUSE-ppds-dat cups-drivers cups-drivers-stp

Tabelle 4.1: Paketübersicht

Die Paketübersicht zeigt, dass die Liste der zu installierenden Pakete lang ist, was aber hauptsächlich daran liegt, dass die Ansteuerung des Druckers nicht das Problem in einer Linux-Umgebung darstellt. Die meisten Linux-Programme nutzen Postscript als Ausgabeformat. Dieses Format wird aber nicht von allen Druckern direkt verstanden: CUPS integriert daher viele Tools, die aus den Postscript-Dateien ein Format generieren, das der Drucker versteht. Die dafür notwendigen Informationen, welches Tool wie für die Konfiguration genutzt werden muss, sind in den sogenannten PPD-Dateien enthalten. Mit den oben genannten Paketen installieren Sie i. d. R. alle bekannten Druckertypen, sodass Sie bei der Konfiguration von CUPS bereits Ihren Drucker finden sollten. Fehlt dieser dort, lohnt sich ein Blick auf die

Webseite www.linuxprinting.org. Dort finden Sie eine Datenbank, die alle zurzeit unter Linux lauffähigen Drucker enthält und die passenden PPD-Dateien generieren kann.

4.1 Einen Drucker einrichten

Nach der Installation können Sie den CUPS-Server über verschiedene Tools konfigurieren; CUPS ist z. B. sehr gut in KDE integriert. Zusätzlich können Sie CUPS über eine Webschnittstelle konfigurieren, was insbesondere dann praktisch ist, wenn der Rechner ein Server ohne grafische Konsole ist. Wir werden Ihnen die Konfiguration mit Hilfe der Webschnittstelle jetzt kurz näherbringen.

Starten Sie den CUPS-Server über sein Init-Skript, wenn er nicht schon läuft. Starten Sie dann einen Webbrowser, und rufen Sie die Adresse http://localhost:631 auf. Wenn Sie auf den Rechner nur über das Netzwerk zugreifen können, müssen Sie an dieser Stelle zu Abschnitt 4.3 auf S. 85 vorblättern, um einen externen Zugriff zu erlauben.

Jetzt können Sie über den Link DRUCKER VERWALTEN neue Drucker einrichten. Da dies eine Verwaltungsaufgabe ist, die nicht allen Nutzern zugänglich sein sollte, fragt der Server Sie nach einer Kennung: Hier kommen Sie mit der root-Kennung des Rechners und dem dazugehörigen Passwort weiter.

Die Einrichtung eines Druckers geschieht in fünf Schritten: Nachdem Sie den Button DRUCKER HINZUFÜGEN angeklickt haben, geben Sie die Informationen zum Dru-

Abbildung 4.1: Startseite des CUPS-Servers

cker ein, die die Nutzer hinterher sehen werden: Der *Name* sollte ein kurzer String ohne Leerzeichen und Sonderzeichen sein. Unter diesem Namen wird der Drucker später angesprochen. Passende Beispiele wären für z. B. einen HP Laserjet 8150 der Name »lj8150« oder aber auch ein eher organisatorischer Name wie »abt2.3.5-li«.

Lachen Sie nicht über die Möglichkeit, einen *Standort* anzugeben: Wenn Sie erstmal 25 Drucker zu verwalten haben, freuen Sie sich, wenn Sie wissen, wo Sie hinlaufen müssen, um Papier nachzulegen. Hier könnte also so etwas wie »Abteilung 2.5.3, 2. Stock, Druckerraum links« stehen. Die *Beschreibung* sollte den Druckertyp etwas genauer enthalten (z. B. »HP Laserjet 8150«), vor allem, wenn Sie vorher einen organisatorischen Namen gewählt haben, an dem man nicht erkennen kann, um was für einen Drucker es sich handelt.

4.1.1 Druckerschnittstelle festlegen

Jetzt kommen wir zu den spannenden Einträgen: der Druckerschnittstelle. Hier definieren Sie, wohin gedruckt wird: ob über eine parallele Schnittstelle, über USB oder über das Netzwerk. Bei parallelen und USB-Schnittstellen sollte in der Auswahlliste bereits zu sehen sein, ob dort ein Drucker erkannt wurde. Dieser muss dafür natürlich angeschlossen sein.

Bei den Netzwerkschnittstellen geben Sie zuerst den Typ an und wählen danach die URL, unter der der Server angesprochen wird:

LPD/LPR-Host or Printer: Der **LPR** mit der Server-Software `lpd` ist das ursprüngliche Drucksystem, das von den Unix-Ahnen überliefert wurde. Neben einer neueren Variante **LPRng**, die als Alternative zu CUPS eingesetzt werden kann, findet man einfache LPR-Installationen häufig auf Netzwerkdruckern, die mit einer eigenen Netzwerkschnittstelle versorgt sind. Wenn Sie also Drucker-Queues auf anderen Unix-Hosts nutzen wollen oder einen Netzwerkdrucker haben, der einen LPR-Service bereitstellt, wählen Sie diese Option und geben die Adresse in der Form `lpd://hostname/queue` an.

AppSocket/HP JetDirect: Netzwerkdrucker von HP bieten, wenn sie netzwerkfähig sind, den Zugriff über ein eigenes Protokoll. Außerdem gibt es von HP diese kleinen Kisten mit bis zu vier parallelen Schnittstellen, mit denen Sie Drucker im Netzwerk bereitstellen können. Diese arbeiten ebenfalls mit dem Protokoll »JetDirect«. Angesprochen werden diese über die Adresse `socket://hostname`. Hier können Sie über verschiedene Portnummern die unterschiedlichen Schnittstellen ansprechen. Keine Portangabe entspricht der Portnummer 9100 und damit der ersten Schnittstelle.

Internet Printing Protocol: CUPS-Server verständigen sich untereinander über das Internet Printing Protocol (IPP). Handelt es sich beim Gegenüber um ein weiteres CUPS-System, sollten Sie die Adresse mit der passenden Protokollspezifikation `ipp://hostname/ipp/druckername` verwenden. Der Eintrag von Druckern anderer CUPS-Server ist jedoch nur notwendig, wenn diese die von ihnen verwalteten Drucker nicht im Netzwerk propagieren dürfen. Mittlerweile existieren auch Drucker, die über IPP angesprochen werden können, nur lauschen diese auf dem Standard-

HTTP-Port. Für diese müssen Sie daher »Internet Printing Protocol (http)« wählen und später die Adresse `http://hostname/` verwenden.

Windows Printer via Samba: Auf unter Windows freigegebene Drucker können Sie ebenfalls drucken. Dazu müssen aber auf dem Linux-Client die Pakete der Samba-Clients installiert sein. Damit können Sie über die Adresse `samba://user:passwort@hostname/sharename` den freigegebenen Drucker ansprechen. Vor allem auf den NT-basierten Windows-Systemen ist die Angabe von Kennung und Passwort wichtig.

4.1.2 Wahl des Druckertyps

In den nächsten beiden Schritten legen Sie dann den Typ des Druckers fest. Wenn Ihnen die Auswahl der Gerätetypen zu gering ist, sollten Sie die foomatic-Pakete nachinstallieren, da diese die verschiedenen PPD-Dateien zur Verfügung stellen, aus denen Sie dann hier eine Auswahl treffen können. Sie müssen die Konfiguration dafür nicht abbrechen. Nehmen Sie einfach einen Dummy-Eintrag oder konfigurieren Sie den Drucker als »Raw«, installieren Sie dann die Pakete, und modifizieren Sie dann später die Druckerkonfiguration des unvollständig eingetragenen Druckers über den Button MODIFIZIEREN.

Damit haben Sie einen Drucker eingerichtet, dessen Funktionsfähigkeit Sie jetzt mit einer Testseite prüfen können. Dazu finden Sie im Bereich DRUCKER VERWALTEN zu jedem dort aufgelisteten Drucker einen Button DRUCKER TESTSEITE.

4.2 Externer Zugriff auf die Webschnittstelle von CUPS

Widmen wir uns noch der Konfigurationsdatei des CUPS-Servers `/etc/cups/cupsd.conf`: In der Standardeinstellung dürfen nur Zugriffe über die Loopback-Schnittstelle Änderungen an den Konfigurationen der eingerichteten Drucker vornehmen. Für eine Remote-Administration wäre es natürlich angenehmer, aus einem definierten Bereich des Subnetzes einen direkten Zugriff auf die Webschnittstelle zu haben.

Die Konfigurationsoptionen lehnen sich an denjenigen an, die in einer Apache-Konfiguration benutzt werden. Sie finden im unteren Teil der Datei Einträge, die mit `<location>` beginnen und jeweils für verschiedene URLs unterschiedliche Zugriffe erlauben. So ist z. B. die Startseite mit folgenden Einstellungen konfiguriert:

```
<Location />
Order Deny,Allow
Deny From All
Allow From 127.0.0.1
Allow From 192.168.1.*
</Location>
```

Wir haben die Konfiguration um eine weitere `Allow`-Zeile ergänzt, in der wir unser Subnetz eingetragen haben. Weiter unten in der Konfigurationsdatei finden

Sie einen weiteren Location-Eintrag, der sich auf den Administrationsbereich bezieht:

```
<Location /admin>
AuthType Basic
AuthClass System

Order Deny,Allow
Allow From 127.0.0.1
Allow From 192.168.1.*
Deny From All
</Location>
```

Auch hier haben wir einen Eintrag für unser Subnetz hinzugefügt. Damit die Änderungen wirksam werden, müssen Sie den CUPS-Dämon neu starten. Jetzt sollten Sie über das Netzwerk mit der IP-Adresse des CUPS-Servers darauf zugreifen können. Hat der Server, auf dem CUPS läuft, die IP-Adresse 192.168.1.10, dann rufen Sie im Webbrowser die Adresse `http://192.168.1.10:631/` auf.

4.3 Netzwerktransparentes Drucken mit CUPS

Im Verein mit Linux-Servern erweist sich das Drucken im Netzwerk als eine einfache Angelegenheit. CUPS verfügt von Haus aus über einen Service, der – einmal aktiviert – Ihnen die Kopfschmerzen der Druckereinrichtung auf Ihren Linux-Clients auf sehr einfache Art und Weise nimmt: die im Netzwerk installierte CUPS-Software kommuniziert miteinander und gibt die auf den jeweiligen Rechnern installierten Drucker bekannt.

Damit es hier nicht drunter und drüber geht, müssen die Druckerfreigaben natürlich freigegeben werden. Dazu werfen wir kurz einen Blick auf die Konfiguration von CUPS auf dem Druckerserver: In der Datei `cupsd.conf` findet sich ein Bereich, der mit `Browsing Options` überschrieben ist. Dort können Sie einstellen, ob CUPS-Dämonen auf anderen Rechnern Druckdienste auf diesem Server nutzen können.

Listing 4.1: cupsd.conf

```
########
######## Browsing Options
########

#
# Browsing: whether or not to broadcast and/or listen for CUPS printer
# information on the network.  Enabled by default.
#

#Browsing On
```

```
...
BrowseAddress 192.168.1.255
...
BrowseAllow 192.168.1.*
BrowseDeny All
...
BrowseOrder deny,allow
```

Wichtig sind die Angaben, in welchem Subnetz die Drucker propagiert werden sollen (`BrowseAddress`) und welche Rechner das Angebot sehen können (`BrowseAllow`). Das »Browsing« ist automatisch angeschaltet; welche Rechner es nutzen dürfen, ist jedoch i. d. R. nicht angegeben. Hier wird im Übrigen nur festgelegt, welche Drucker gesehen werden können. Auf welche dann effektiv gedruckt werden darf, wird unter der Überschrift `Security Options` eingetragen; wenn man es möchte, sogar für jeden Drucker einzeln. Wir zeigen Ihnen einmal ein Beispiel für den Drucker `abt2.3.5-li` mit einer eigenen Deklaration:

```
########
######## Security Options
########
...
<Location /printers/abt2.3.5-li>
## Anonymous access (default)
AuthType None
...
Order Deny,Allow
Deny From All
Allow From localhost
Allow From 192.168.1.*
</Location>
```

Sie können mit Hilfe mehrerer `Allow From`-Zeilen auch mehrere Einzel-IP-Nummern oder mehrere Subnetze eintragen.

Die CUPS-Dämonen beginnen dann, mit Hilfe von Broadcasts ihre Anwesenheit im Netzwerk kundzutun. Sobald Sie also einen Client starten und dort CUPS hochgefahren wird, sollte nach einer kurzen Weile der auf dem Server eingerichtete Drucker auch auf dem Client zu finden sein:

```
marcus@theodor:~$ lpstat -t
scheduler is running
system default destination: abt2.3.5-li
device for abt2.3.5-li: ipp://hugo:631/printers/abt2.3.5-li
abt2.3.5-li accepting requests since Jan 01 00:00
printer abt2.3.5-li now printing abt2.3.5-li-0.  enabled since Jan 01 00:00
marcus@theodor:~$
```

Da unter Linux alles Druckwerk das Postscript-Format besitzt und auch PDF als Postscript-Ableger ohne Probleme direkt an die Druckausgabe geschickt werden kann, müssen Sie auf den Clients auch keine weiteren Druckertreiber installieren. Da auf dem Rechner, an dem der Drucker physikalisch angeschlossen ist, die Umwandlung in die jeweilige Druckersprache vorgenommen wird, können die Druckdateien im Postscript-Format zwischen Client und Server ausgetauscht werden.

Achtung

Vergessen Sie bei SUSE und Fedora nicht, den ipp-Port 631 in der Firewall freizuschalten, und zwar sowohl für TCP- als auch für UDP-Pakete. Wenn die Firewall die Broadcast-Pakete der im Netz befindlichen CUPS-Server blockt, funktioniert weder das Drucken noch die Propagierung der eingerichteten Drucker.

4.4 Weitere Tipps zur Konfiguration von CUPS

Die Standard-Konfiguration von CUPS lässt i. d. R. Änderungen nur mit Hilfe der root-Kennung zu. Sie können jedoch auch Nutzerkennungen die Berechtigung geben, Drucker zu verwalten. In der Konfigurationsdatei existiert ein Eintrag `SystemGroup`, der auf Debian-Systemen auf die Gruppe `lpadmin` konfiguriert ist. Der Gruppenname bezieht sich auf die Gruppen, die auf der Linux-Ebene, also in der Datei `/etc/group` definiert sind. Kennungen, die Sie dort hinzufügen, können sich mit dem zur Kennung gehörenden Passwort anmelden und haben dann die gleichen Rechte wie die root-Kennung im Bezug auf die Druckerverwaltung.

Webzugriff über eine sichere SSL-Verbindung: Standardmäßig läuft der Zugriff auf die Webschnittstelle von CUPS nicht verschlüsselt ab. Um eine geschützte Übertragung zu aktivieren, muss in der `cupsd.conf` bei den oben genannten Location-Einträgen die Option `Encryption Required` eingetragen sein. Dies allein reicht jedoch noch nicht aus; es müssen noch Zertifikate für die Verschlüsselung vorhanden sein. Diese sind i. d. R. jedoch noch nicht angelegt (Näheres zur Erzeugung eigener SSL-Zertifikate finden Sie in Abschnitt 9.6.2 und Abschnitt 11.2.4). Wenn Sie bereits ein SSL-Zertifikat besitzen, können Sie dies im Prinzip für den CUPS-Dämon wiederverwenden:

```
ServerCertificate /etc/ssl/certs/local.crt
ServerKey /etc/ssl/private/local.pem
...
Encryption Required
</Location>
```

Über die Optionen `ServerCertificate` und `ServerKey` teilen Sie dem CUPS-Dämon mit, wo sich Zertifikat und privater Schlüssel befinden. Wenn Sie dann für die Location

`/admin Encryption` auf `Required` setzen, können Sie über die Adresse `https://192.168.1.10:631/admin` eine verschlüsselte Verbindung zur Webschnittstelle aufbauen.

Debian-Tipp

Bei Debian ist der CUPS-Dämon nicht mit der Möglichkeit kompiliert worden, SSL zu nutzen. Daher können Sie diese Optionen hier zwar eintragen, der SSL-Zugriff bricht aber mit einer Fehlermeldung ab. Da bei der unverschlüsselten Nutzung der Webschnittstelle die Passwörter im Klartext über das Netz gehen, empfehlen wir hier unbedingt, eine andere Kennung als die root-Kennung zur Drucker-Konfiguration zu nutzen. Sie können aber auch mit Hilfe von `stunnel` eine SSL-Verbindung vorschalten.

Und zum Schluss noch ein Hinweis: Ab Windows 2000 können Windows-Rechner CUPS auch direkt per IPP-Protokoll ansprechen! Dazu erstellen Sie diesen Drucker als Netzwerkdrucker und geben als URL z. B. `http://192.168.1.10:631/Printers/abt2.3.5-li` an. Als Druckertreiber können Sie dann sowohl Postscript als auch den zum Drucker gehörenden Treiber auswählen. Der CUPS-Dämon erkennt automatisch, ob es sich um Postscript oder bereits aufbereitete Druckdaten handelt.

5 Automatische Vergabe von IP-Adressen mit DHCP

Die automatische Vergabe von IP-Adressen ist (fast) so alt wie das Netzwerken überhaupt. Die Anfänge machte das Protokoll BOOTP, das neben der Vergabe von IP-Adressen auch in der Lage war, eine Datei zu übertragen, mit der festplattenlose Clients booten konnten. Mittlerweile übernimmt das DHCP-Protokoll diese Aufgabe mit einem leicht erweiterten Funktionsumfang.

Notwendig ist eine DHCP-Serversoftware, die als Dämon auf einem Rechner läuft, der sich physikalisch in dem Subnetz befindet, in dem sich auch die zu betreuenden Clients befinden. DHCP-Anfragen werden i. d. R. von Routern nicht weitergeleitet, da es sich dabei um Broadcast-Requests handelt, die nicht unkontrolliert in die Weiten des Internets gelangen dürfen.

Dem DHCP-Server wird per Konfiguration mitgegeben, auf welcher Netzwerkschnittstelle er auf DHCP-Broadcasts reagieren soll. Welche bei ihm ankommen, wird durch die Netzwerktopologie bestimmt. Die Konfigurationshilferufe des Clients dürfen also auf dem Weg zum Server von keiner Netzwerkkomponente blockiert werden.

DHCP-Server	Debian	Fedora	SUSE
Standard-DHCP-Server Der vom Internet Software Consortium bereitgestellte DHCP-Server in der Version 3	dchp3-server	dhcp	dhcp-server
DHCP-Clients Verschiedene Clients, mit denen man unter Linux per DHCP IP-Adressen beziehen kann	dchp3-client dhcpcd pump	dhclient	dhclient dhcpcd
DNSMasq mit DHCP-Funktionen Die Software DNSMasq stellt ebenfalls DHCP-Funktionen bereit	dnsmasq	dnsmasq	dnsmasq

Tabelle 5.1: Paketübersicht

Wir erläutern die Bereitstellung von DHCP-Funktionen am Beispiel des Standard-DHCP-Servers. Die Software DNSMasq kann aber auch DHCP-Requests bedienen. Näheres dazu finden Sie in Abschnitt 3.5. Wie man von einem Linux-Client aus per DHCP an eine Adresse herankommt, haben wir bereits in Abschnitt 2.1.3 beschrieben.

Machen wir uns zuerst einmal klar, welche Daten wichtig für den Client sind, d. h., welche Parameter wir zentral vergeben wollen:

- Die *IP-Nummer*: diese muss aus dem Bereich unseres Subnetzes vergeben werden. Wir müssen uns natürlich klar darüber sein, dass wir den IP-Bereich, den wir dem DHCP-Server zur Verfügung gestellt haben, nicht anderweitig verwenden können. Daher behalten wir noch ein paar IP-Nummern übrig und geben nur 192.168.1.100 bis 192.168.1.250 für die dynamische Vergabe frei.
- Die IP-Nummer des *Gateways* (192.168.1.1), über das die Clients auf das Internet zugreifen können,
- Den oder die IP-Nummern der *Nameserver* (wir verwenden in unseren Beispielen 192.168.1.1 und 192.168.1.5), damit auch die Hostnamen entsprechend umgesetzt werden können.

Bei der Installation der DHCP-Serversoftware wird eine Beispiel-Konfiguration in der Datei `dhcpd.conf` angelegt. Am besten legen Sie von dieser Datei eine Kopie an, bevor Sie dort Änderungen vornehmen. Die Kommentare in der Datei sind eine erste Quelle, die die Bedeutung der einzelnen Einträge erläutern. Eine Referenz aller Optionen, die in der Konfigurationsdatei genutzt werden können, liefert `man dhcpd.conf`.

Debian	Ab der Version 3 wird für den DHCP-Server in `/etc` ein eigenes Unterverzeichnis `dhcp3` angelegt. Dort finden Sie auch die Konfigurationsdatei `dhcpd.conf`. Die Schnittstellen, die der DHCP-Server bedienen soll, werden in der Datei `/etc/default/dhcp3-server` eingetragen.
SUSE	SUSE bietet für die Konfiguration des DHCP-Services ein YaST-Modul an, das Sie auf der Kommandozeile mit `yast dhcp-server` starten können. Dort werden die einzugebenden Daten formularbasiert abgefragt. Denken Sie daran, die Firewall für die Schnittstelle zu öffnen, auf der der DHCP-Server Anfragen beantworten soll! Sie können die Daten aber auch mit dem Editor in die Konfigurationsdateien eintragen: `dhcpd.conf` liegt im Verzeichnis `/etc`. Zusätzlich stellen Sie in `/etc/sysconfig/dhcpd` die Schnittstellen ein, auf denen der DHCP-Server lauschen soll.

Tabelle 5.2: Unterschiede zwischen den Distributionen

Beginnen wir mit der Konfiguration des Subnetzes, für das IP-Nummern vergeben werden sollen:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.1;
  option domain-name-servers 192.168.1.1, 192.168.1.5
  range 192.168.1.100 192.168.1.250;
}
```

Mit dem Schlüsselwort `subnet` wird ein Abschnitt in der Konfigurationsdatei eingeleitet, der die für das Subnetz spezifischen Informationen enthält. Diese werden innerhalb der geschweiften Klammern über das Schlüsselwort `option` eingetragen.

Wichtig ist das Semikolon, das am Ende jeder Zeile stehen muss. Mehrere Einträge können durch Kommata getrennt eingegeben werden, wie Sie am Beispiel der Einträge zu den DNS-Servern sehen können, die über die Option `domain-name-server` festgelegt werden. Nach den Optionen werden die sogenannten Deklarationen eingetragen. Dort haben wir im Moment nur eine Angabe zu machen, den Bereich (`range`), aus dem die zu verteilenden IP-Nummern genommen werden sollen – an dieser Stelle ausnahmsweise ohne Kommata.

5.1 Leasing-Zeiten für IP-Nummern

Mit der Eintragung des Subnetzes haben wir die für uns wichtigen Einträge gemacht. Der DHCP-Server hätte aber gern noch ein paar Angaben darüber, wie er sich in Extremfällen verhalten soll. Eine dieser Angaben bezieht sich auf den Zeitraum, über den eine IP-Nummer für einen Rechner reserviert ist, die sogenannte **Leasing-Zeit**. Mit dieser Zeitspanne legen Sie fest, wie lange der DHCP-Server nach der Anfrage des Clients eine Adresse reserviert. Läuft diese Zeitspanne ab – und sorgt der Client nicht für eine Auffrischung, weil er z. B. abgeschaltet ist –, wird diese IP wieder in den Pool der freien IP-Nummern aufgenommen.

Hier lassen sich verschiedene Szenarien vorstellen, in denen unterschiedlichen Leasing-Zeiten eine wichtige Bedeutung zukommt. IP-Nummern sind mittlerweile ein rares Gut geworden; die Zahl der Computer, die an das Internet angeschlossen werden sollen, liegt in Einzelfällen, in denen ein Unternehmen oder eine Einrichtung nur einen kleinen IP-Bereich zugewiesen bekommen hat, deutlich über der Anzahl der vergebbaren IP-Nummern.

Ein Szenario stellt ein WLAN-Hotspot dar: Dort sollen viele verschiedene Computer für kurze Zeit eine IP-Nummer zugewiesen bekommen. Anders ist die Situation in einem Bereich, in dem die Anzahl der Computer feststeht, die über DHCP eine IP-Nummer zugewiesen bekommen sollen. Dort ist es wünschenswert, dass ein Rechner, wenn er einmal eine IP-Nummer bekommen hat, diese auch möglichst lange behält. Wenn man hier keine festen IP-Nummern vergeben möchte, kann man dieses Problem über eine längere Leasing-Zeit regeln.

```
default-lease-time 86400;     # 1 Tag
max-lease-time 259200;        # 3 Tage

subnet 192.168.1.0 netmask 255.255.255.0 {
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.1;
  option domain-name-servers 192.168.1.1, 192.168.1.5
  range 192.168.1.100 192.168.1.250;
}
```

Die Konfiguration erlaubt die Angabe von zwei Werten: Zum einen die Angabe einer `default-lease-time`, die Clients mitgeteilt wird, die nicht nach einer speziellen Ablaufzeit fragen. Zum anderen kann mit `max-lease-time` noch eine maximale

Zeit festlegen, die eine IP-Adresse einem Rechner zugeordnet wird. Wenn Sie sicher gehen wollen, dass genau eine Leasing-Zeit den Rechnern mitgegeben wird, sollten Sie beide Zeiträume auf den gleichen Wert setzen. Die Angaben erfolgen in *Sekunden*! Für längere Zeiträume wie in unserem Beispiel oben ist daher etwas »Rechenleistung« notwendig. :-)

5.2 Kontrolle des DHCP-Servers

Jetzt ist es an der Zeit, die eben erzeugte Konfiguration zu testen. Starten Sie den DHCP-Server über sein Init-Script (bei SUSE und bei Fedora `/etc/init.d/dhcpd`), und schauen Sie danach mit `tail -f` in das Syslog (bei SUSE und Fedora `/var/log/messages`):

```
hugo:~# /etc/init.d/dhcp3-server start
hugo:~# tail -f /var/log/syslog
Xxx 13 18:32:38 hugo dhcpd: Internet Systems Consortium DHCP Server V3.0.1
Xxx 13 18:32:38 hugo dhcpd: Copyright 2004 Internet Systems Consortium.
Xxx 13 18:32:38 hugo dhcpd: All rights reserved.
Xxx 13 18:32:38 hugo dhcpd: For info, please visit http://www.isc.org/sw/dhcp/
Xxx 13 18:32:38 hugo dhcpd: Wrote 0 deleted host decls to leases file.
Xxx 13 18:32:38 hugo dhcpd: Wrote 0 new dynamic host decls to leases file.
Xxx 13 18:32:38 hugo dhcpd: Wrote 0 leases to leases file.
```

Im Syslog werden die Anfragen und Vergaben protokolliert. Außerdem meldet der Server hier eventuell auftretende Probleme. Jetzt fehlt uns für unseren Test nur noch ein Rechner, der eine DHCP-Anfrage stellt. Wir haben einen, haben Sie auch einen? Dann schalten Sie ihn einmal an, oder lassen auf ihm die DHCP-Abfrage laufen:

```
Xxx 13 19:03:00 hugo dhcpd: DHCPDISCOVER from fe:fd:00:00:00:aa via eth0
Xxx 13 19:03:01 hugo dhcpd: DHCPOFFER on 192.168.1.100 to fe:fd:00:00:00:aa via eth0
Xxx 13 19:03:01 hugo dhcpd: DHCPREQUEST for 192.168.1.100 (192.168.1.10) from fe:fd:
00:00:00:aa via eth0
Xxx 13 19:03:01 hugo dhcpd: DHCPACK on 192.168.1.100 to fe:fd:00:00:00:aa via eth0
 STRG + C
hugo:~#
```

Wenn alles klappt, sehen Sie diese vier Einträge: Der Rechner, der gerade die Anfrage gestellt hat, meldet sich und wird anhand der MAC-Adresse identifiziert. Wenn ihm noch kein anderer Rechner zuvorgekommen ist, erhält er die erste IP-Nummer aus dem oben definierten Adressraum. Wenn Sie genug gesehen haben, können Sie mit STRG + C die Ausgabe des Syslog beenden.

Wenn der DHCP-Server nicht starten will, also das Init-Script mit einer Fehlermeldung abbricht, liegt es i. d. R. an einem Fehler in der Konfigurationsdatei. Sie können dann im Syslog nachsehen, welcher Eintrag hier Probleme macht (bei Fedora und SUSE werden diese Informationen direkt auf der Kommandozeile ausgegeben):

```
hugo:~# /etc/init.d/dhcpd start
Starting DHCP server Internet Systems Consortium DHCP Server V3.0.1
Copyright 2004 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
/var/lib/dhcp///etc/dhcpd.conf line 7: expecting numeric value.
subnet  netmask
        ^
Configuration file errors encountered~-- exiting
# Und noch eine ganze Menge mehr Text ...
hugo:~#
```

Bei SUSE ist es uns passiert, dass nach der Konfiguration mit YaST hier die passenden Einträge fehlten. Und der DHCP-Server ist da eigen, ohne will er nicht. Als wir die Werte per Hand nachgetragen hatten, konnten wir ihn aber schnell dazu bewegen, seine Arbeit aufzunehmen. Er weist uns ja auch freundlich darauf hin, in welcher Zeile wir nach dem Fehler zu suchen haben.

Weitere Literatur

- `man dhcpd.conf`: Die Manpage ist eine gute Referenz mit einer ausführlichen Dokumentation aller Optionen und mit Beispielen.
- »DHCP-Mini-Howto«: Für den schnellen Einstieg. Sie finden es entweder schon auf Ihrem Rechner oder unter `http://www.tldp.org/HOWTO/DHCP`

5.3 Mit DHCP feste IP-Nummern vergeben

Wenn man beginnt, größere Netzwerke zu verwalten, und eine größere Herde von Arbeitsplatzrechnern bewachen muss, ist es i. d. R. lästig, wenn diese mit dynamischen IP-Nummern arbeiten. Den Rechnern kann dann kein fester Name zugeordnet werden, und seltsame Phänomene wie z. B. Portscans auf Grund von Virenbefall lassen sich nur zuordnen, wenn man alle Rechner abklappert, um herauszufinden, wer gerade welche IP-Nummer hat. Ein erster Schritt wäre, eine Liste mit Hardware-Adressen zu führen. Der DHCP-Server führt die Liste der vergebenen Adressen in der Datei `/var/lib/dhcp3/dhcpd.leases`. Dies ist eine auch für Menschen lesbare Textdatei, in der Sie die aktuelle Zuordnung der IP zu einer MAC-Adresse finden können.

Der sicherheitsbewusste Netzwerkadministrator wird Letzteres sowieso tun – auch wenn es unter Umständen nicht viel nutzt, weil die IP-Nummern fest vergeben worden sein könnten, siehe Abschnitt A.1.3 – und in diesem Fall sollte man die Möglichkeiten von DHCP nutzen, um sich das Leben einfacher zu machen. Es ist nämlich auf der Basis der MAC-Adressen ohne weiteres möglich, feste IP-Nummern zu vergeben.

Zuerst sollten wir uns jedoch Gedanken darüber machen, wie wir unser Netzwerk aufbauen wollen, und dann in welchen Größenverhältnissen die Anzahl der fest

und die der dynamisch zu vergebenden IP-Nummern liegen: Daraus ergibt sich dann, wie wir unseren IP-Bereich aufteilen müssen. Und dann wäre da noch die Frage, welche Rechner man in die dynamische Vergabe einbeziehen sollte und welche nicht.

Stellen wir uns also vor, wir hätten ein paar Server, die uns intern mit Webseiten, Mail und sonstigen Dingen beliefern; dann eine Reihe von Arbeitsplatzrechnern, die an festen Plätzen unter oder auf Schreibtischen stehen; außerdem haben wir noch einen Bereich, an dem sich unsere dynamischen Außendienstmitarbeiter mit ihren Laptops über einen WLAN-Hotspot einloggen.

- **Welche Rechner sollten über DHCP mit IP-Nummern bedient werden?** Es ist i. d. R. keine gute Idee, Server, die mit festen Aufgaben betraut sind, von der Arbeit eines DHCP-Servers abhängig zu machen. Bei diesen Servern sollten Sie die IP-Nummer fest eintragen. Die Arbeitsplatzrechner in unserer Filiale sind jedoch ein gutes Opfer für die feste Vergabe einer IP-Nummer per DHCP. Der WLAN-Hotspot dient auch Mitarbeitern, die aus anderen Filialen kommen und bei uns hereinschneien, um sich hier ins Firmennetzwerk einzuwählen. Daher haben wir bei diesen keine Übersicht über die verwendeten Netzwerkkarten und damit über die MAC-Adressen. Wir weisen diesen Rechnern also dynamische Adressen zu.
- **Wie viele Adressen für welchen Bereich**? Wir sind eine kleine Filiale, daher ist nicht damit zu rechnen, dass mehr als 50 Arbeitsplatzrechner in unserem Haus aufgestellt werden. Wir haben zwar nicht so viele Server, aber es kann unter Umständen (je nach Ausfall-Konzept) sinnvoll sein, Funktionen wie Router, DNS-Server, Mailserver und Webserver mit spezifischen IP-Adressen zu verbinden und so einem Server mehrere Adressen zuzuweisen.
- Wir entscheiden uns daher, die IP-Nummern von `192.168.1.1` bis `192.168.1.50` aus der Verteilung durch DHCP herauszunehmen. Für die Vergabe der festen IP-Nummern reservieren wir dann den Bereich von `192.168.1.100` bis `192.168.1.150` und legen den dynamischen Bereich mit `192.168.1.200` bis `192.168.1.230` fest. Wenn jetzt im internen Netz ein Missbrauch oder eine Störung auftritt, können wir schon anhand der IP-Nummer zumindest erkennen, bei welcher Art von Rechnern der Fehler zu suchen ist.

Unsere Konfigurationsdatei sieht nach diesen Überlegungen so aus:

Listing 5.1: dhcpd.conf

```
default-lease-time 86400;     # 1 Tag
max-lease-time 259200;        # 3 Tage

subnet 192.168.1.0 netmask 255.255.255.0 {
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.1;
  option domain-name-servers 192.168.1.1, 192.168.1.5
  pool {
```

```
    deny unknown-clients;
    range 192.168.1.100 192.168.1.150;
  }
  pool {
    max-lease-time 120;
    range 192.168.1.200 192.168.1.230;
    allow unknown-clients;
  }
}
```

Wir können mit dem Schlüsselwort `pool` in einer Subnetz-Definition Pools von IP-Nummern definieren und diesen spezielle Optionen zuweisen. Aus dem ersten Pool werden z. B. keine IP-Nummern an unbekannte MAC-Adressen (`deny unknown-clients`) vergeben. Im zweiten Pool – dem für die WLAN-Hotspots mit den häufig wechselnden Clients – haben wir die Leasing-Zeit auf eine sehr kurze Zeitspanne gesetzt, damit eine IP-Nummer nicht unnötig lange für einen einzelnen Client reserviert wird.

Jetzt fehlt nur noch die Eingabe der festen IP-Nummern: Dazu legen wir für jeden Arbeitsplatzrechner im ersten Pool einen Eintrag in der folgenden Form an:

Listing 5.2: dhcpd.conf

```
...
  pool {
    deny unknown-clients;
    range 192.168.1.100 192.168.1.150;
    host theodor  hardware ethernet FE:FD:00:00:00:AA; fixed-address 192.168.1.100;
  }
...
```

Hinter dem Schlüsselwort `host` muss immer ein Hostname eingetragen werden. Wenn dieser Name für den DHCP-Server in eine IP-Adresse aufgelöst werden kann, kann die Angabe `fixed-address 192.168.1.100;` weggelassen werden. Hinter `hardware ethernet FE:FD:00:00:00:AA` verbirgt sich die MAC-Adresse des Clients, wie Sie sich sicher schon gedacht haben.

Achtung

Bedenken Sie, dass der DHCP-Server mit dieser Konfiguration nicht erkennen kann, woher die Anfrage kommt oder von was für einem Typ von Rechner. Alle Arbeitsplatzrechner bekommen über die dynamische Vergabe eine IP-Nummer, solange sie keinen `host`-Eintrag im Pool mit den festen IP-Nummern haben!

5.4 DHCP in verschiedenen Subnetzen

Woran erkennt der DHCP-Server, in welchem Subnetz er sich befindet? Die Antwort liegt in der IP-Nummer der Schnittstellen, auf denen er nach Anfragen lauschen soll. Wenn `eth0` die IP-Nummer 192.168.1.10 zugewiesen wurde, dann weist der DHCP-Server Anfragen, die über `eth0` hereinkommen, eine IP-Nummer aus der Konfiguration für das Subnetz 192.168.1 zu. Daraus ergibt sich eine einfache Schlussfolgerung: Mit den bisherigen Konfigurationsoptionen lassen sich verschiedene Subnetze nur über unterschiedliche Netzwerkschnittstellen betreuen; ein logisches Subnetz entspricht in diesem Fall einem physikalischen Subnetz.

Ein Linux-Rechner, der mit mehreren Netzwerkkarten ausgerüstet ist, kann über jede der Netzwerkkarten ein anderes Subnetz mit IP-Nummern füttern. Wenn jetzt aber die Anzahl an IP-Nummern eines Subnetzes für die zu versorgenden Client-Rechner nicht mehr ausreicht, kann der Server mit der Direktive `shared-network` darüber informiert werden, dass mehrere Subnetze zu einem physikalischen Netzverbund gehören. Hier gilt natürlich auch, dass alle Broadcast-Anfragen der suchenden Clients zum DHCP-Server gelangen können müssen.

Listing 5.3: dhcpd.conf

```
default-lease-time 86400;     # 1 Tag
max-lease-time 259200;        # 3 Tage

shared-network gesamt {
  option domain-name-servers 192.168.1.1, 192.168.1.5
  subnet 192.168.1.0 netmask 255.255.255.0 {
    option broadcast-address 192.168.1.255;
    option routers 192.168.1.1;
    deny unknown-clients;
    range 192.168.1.50 192.168.1.250;
    host theodor  hardware ethernet FE:FD:00:00:00:AA; fixed-address 192.168.1.50;
    ...
  }
  subnet 192.168.2.0 netmask 255.255.255.0 {
    max-lease-time 120;
    option broadcast-address 192.168.2.255;
    option routers 192.168.2.1;
    range 192.168.2.50 192.168.2.250;
    allow unknown-clients;
  }
}
```

In unserem Beispiel haben wir zwei Subnetze über die Option `shared-network` zusammengefasst. Subnetz 192.168.1 ist für die Vergabe fester IP-Nummern reserviert, während dynamisch vergebene IP-Nummern nur aus dem IP-Bereich 192.168.2 vergeben werden. Beachten Sie, dass wir mit der Angabe `option routers 192.168.2.1` davon

ausgehen, dass es ein zweites Gateway im Netzwerk gibt oder der Router jetzt mit zwei IP-Nummern konfiguriert ist. Das erspart komplizierte Konstruktionen bezüglich der Netzmaske.

5.5 Booten über das Netzwerk

Moderne Ethernet-Karten bringen eine Funktion mit, die das Booten des Rechners über das Netzwerk erlaubt. Wir wollen Ihnen an dieser Stelle kurz zeigen, wie Sie mit Hilfe eines laufenden Linux-Rechners die Installation per Netboot durchführen können. Unser Ziel ist es, die Boot-CD zu ersetzen; für die restliche Installation ist dann noch ein Netzwerkzugang oder ein lokaler Mirror notwendig.

TFTP-Server	Debian	Fedora	SUSE
atftp-Server Fortgeschrittene Variante eines TFTP-Servers	atftpd	—	atftp

Tabelle 5.3: Paketübersicht

Das Booten mit Hilfe der PXE-Erweiterung moderner Netzwerkkarten benötigt natürlich einen Server im lokalen Netzwerk, der auf die Anfragen des bootenden Rechners reagiert. Dazu benötigen wir serverseitig

1. einen DHCP-Server, der die Anfrage beantwortet und dem Client, auf dem wir ein Linux installieren wollen, IP-Nummer und Bootinformationen übermittelt,
2. und einen TFTP-Server, der die notwendigen Dateien übermitteln kann.

Der DHCP-Server kann, mit folgender Konfiguration bestückt, bootwillige Clients mit den notwendigen Daten versorgen:

```
default-lease-time 86400;     # 1 Tag
max-lease-time 259200;        # 3 Tage

subnet 192.168.1.0 netmask 255.255.255.0 {
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.1;
  option domain-name-servers 192.168.1.1, 192.168.1.5
  range 192.168.1.100 192.168.1.250;
  filename "/tftpboot/pxelinux.0";
}
```

Wenn Sie bereits einen DHCP-Server im Netzwerk zur Verfügung stellen, brauchen Sie nur die fett gedruckte Zeile `filename ...` nachzutragen.

Worauf weist nun diese Direktive hin? Der Client erhält auf Anfrage vom DHCP-Server neben der IP-Nummer den Hinweis, dass er zusätzlich eine spezielle Datei

herunterladen soll. Nachdem er vom Server die IP-Nummer erhalten hat, versucht er in einem zweiten Schritt, über einen TFTP-Zugriff die angegebene Datei herunterzuladen. Welche er nun herunterladen soll, hängt davon ab, was Sie vom Client erwarten. Wenn Sie eine Debian-Installation über das Netzwerk starten wollen, gibt es dazu ein spezielles Archiv mit dem Namen `netboot.tar.gz`. Dieses sollten Sie im Verzeichnis `/tftpboot` auspacken, sodass das darin enthaltene Verzeichnis `pxelinux.0` auf oberster Ebene zu finden ist. Die dafür notwendigen Dateien können Sie ebenfalls in diesem Verzeichnis ablegen und mit der Option `filename` angeben.

Das Verzeichnis `/tftpboot` stellt das Standard-Verzeichnis dar, das von einem TFTP-Server genutzt wird. Wir empfehlen für das weitere Vorgehen die modernere Variante mit dem Namen `atftpd`. Nach der Installation sollte der TFTP-Server bereits so eingerichtet sein, dass er Dateien aus dem Verzeichnis `/tftpboot` herauszugeben im Stande ist. Eventuell müssen Sie Ihren Inetd-Dämon noch dazu bewegen, sich die Konfigurationsdatei noch einmal anzuschauen, damit er den geänderten Verhältnissen Rechnung tragen kann.

```
hugo:~# fgrep tftpd /etc/inetd.conf
# Um tftp zu nutzen, sollte die nächste Zeile auskommentiert sein.
tftp dgram udp wait  nobody /usr/sbin/tcpd /usr/sbin/in.tftpd /tftpboot
hugo:~# /etc/init.d/inetd reload
hugo:~#
```

Wenn Sie in den Dateien `/etc/hosts.deny` und `/etc/hosts.deny` Anpassungen an die Gegebenheiten in Ihrem lokalen Netzwerk vorgenommen haben, sollten Sie in der Datei `/etc/hosts.allow` noch eine Zeile für den Serverprozess `in.tftpd` nachtragen. Ansonsten sollte der TFTP-Zugriff aktiviert sein.

DHCP-Server gestartet? Dann sollten Sie jetzt einmal versuchen, den Rechner zu starten, der über das Netzwerk booten soll. Sie können das Geschehen auf dem Server beobachten, indem Sie sich während des Bootvorgangs das Syslog ausgeben lassen:

```
hugo:~# tail -f /var/log/syslog
Xxx  7 21:37:11 hugo dhcpd: DHCPACK on 192.168.1150 to 00:00:e2:6b:26:06 via eth0
Xxx  7 21:37:11 hugo in.tftpd[6787]: connect from 192.168.1.150 (192.168.1.150)
Xxx  7 21:37:11 hugo atftpd[6787]: Advanced Trivial FTP server started (0.7)
Xxx  7 21:37:11 hugo atftpd[6787]: Serving /tftpboot/pxelinux.0 to 192.168.1.150:2070
Xxx  7 21:37:11 hugo atftpd[6787]: Serving /tftpboot/pxelinux.0 to 192.168.1.150:2071
...
hugo:~#
```

Das Syslog-Beispiel zeigt, dass unser Client die IP-Nummer 192.168.1.150 zugewiesen bekommen hat und danach versucht, die Datei `/tftpboot/pxelinux.0` herunterzuladen. Auf dem Client merken Sie das gelingen daran, dass Sie einen Boot-Prompt erhalten und damit die Installation starten können.

5.6 Was kann DHCP leisten und was nicht?

Der organisatorische Aufbau Ihres Subnetzes ist i. d. R. keinen großen Veränderungen unterworfen; je besser Sie ihn planen, desto weniger wird sich über einen längeren Zeitraum ändern. Wenn jedoch Änderungen anstehen, kann Ihnen DHCP dabei helfen, diese Änderungen mit geringstmöglichem Aufwand durchzuführen. Nehmen wir an, Sie wollen aus organisatorischen Gründen den lokalen DNS-Server auf einen anderen Rechner verlagern: Bei den Rechnern, die über DHCP konfiguriert werden, müssen Sie nur die jeweiligen Einträge in der Konfiguration ändern. Beim nächsten Neustart – bzw. beim nächsten DHCP-Request werden die Clients die neue Konfiguration nutzen, ohne dass der Anwender etwas von der Änderung merkt.

Trotzdem sollten Sie sich überlegen, bestimmte Funktionen wie Mailservice, Webservice oder Zeitservice auf feste IP-Nummern zu legen und diese über einen einfachen lokalen Nameservice (z. B. DNSMasq, Abschnitt 3.5) mit diesen Funktionen zu verbinden. Den Zugriff auf einen Zeitserver, über den die Clients ihre Uhrzeit synchronisieren können, können Sie z. B. über den Namen *ntp-server* anbieten. Wandert dieser Service auf einen anderen Server, ziehen Name und IP-Nummer mit um und keiner der Rechner, die diesen Service nutzen, müssen umkonfiguriert werden.

DHCP macht Ihr Netzwerk aber nicht sicherer; durch die dynamische Vergabe ist eher das Gegenteil der Fall. Auch die Vergabe fester IP-Nummern mit Hilfe der MAC-Adressen ist kein Garant dafür, dass sich nur Rechner in Ihr Netz einbinden können, über die Sie die Kontrolle haben. Die MAC-Adresse, mit der die Netzwerkkarte hinterher im Netzwerk arbeitet, ist keineswegs so fest »eingebrannt«, dass sie nicht durch einen anderen Wert ersetzt werden kann. Wiegen Sie sich daher nicht in Sicherheit, und nutzen Sie die MAC-Adressen nicht, um uneingeschränkt Privilegien zu verteilen. Dafür sollten zusätzliche Methoden der Autorisierung genutzt werden, die vor allem nicht rechner-, sondern Nutzer-basiert arbeiten. Um aber ein große Anzahl von Rechnern ohne großen Aufwand ins Netz zu bekommen, ist DHCP sicher die beste Wahl.

6 SMB-Netzwerkdienste mit Samba

Interoperabilität haben sich die Entwickler der Software Samba auf die Fahnen geschrieben. Dabei hat der Name nichts mit dem gleichnamigen Tanz zu tun, auch wenn man bei der Vernetzung von Windows-Rechnern mitunter mit dem Wolf tanzen muss. Grundlage der Software sind die Protokolle SMB/CIFS, das **S**erver **M**essage **B**lock-Netzwerk-Protokoll und das **C**ommon **I**nternet **F**ile **S**ystem; Ersteres hat zum Namen von Samba beigetragen. Wir werden Ihnen im Folgenden die Funktionsweise von Samba ab der Version 3 beschreiben. Ältere Versionen verhalten sich u. U. anders bzw. nutzen andere Konfigurationsoptionen.

Samba	Debian	Fedora	SUSE
Samba-Server Samba-Serverpakete	samba	samba	samba
Samba-Clients Clients, um unter Linux auf Windows-Server zugreifen zu können	samba-client	samba-client	samba-client
Konfigurationstools Toolprogramme für Konfiguration von Samba	swat	system-config-samba samba-swat	yast2-samba-client yast2-samba-server
Dokumentation Pakete mit der Samba-Dokumenation	samba-doc	samba-common	samba-doc

Tabelle 6.1: Paketübersicht

Worum handelt es sich bei Samba? Samba stellt Linux-Services für Windows-Clients zur Verfügung, z. B. für das Datei- und Druckersharing. Samba bietet in diesem Zusammenhang aber keine eigenständigen Funktionen, sondern auf Linux-Servern bereits eingerichtete Services an.

Zusätzlich bietet es die Möglichkeit, einen Linux-Server als Domain-Controller für ein Windows-Netzwerk zu nutzen. Widmen wir uns aber zuerst den Basisfunktionen, die die Bereitstellung von Dateien und Druckern beinhalten. Wir werden uns dabei hauptsächlich auf die Konfigurationsdatei `smb.conf` beziehen und Ihnen die dort notwendigen Eintragungen erklären. Im letzten Teil des Kapitels zeigen wir

Ihnen aber auch das Tool SWAT, mit dem Sie die Konfiguration über eine Webschnittstelle vornehmen können.

SUSE-Tipp

SUSE bietet für die Samba-Konfiguration eigene YaST-Module an, die Sie jedoch extra installieren müssen. Die Namen der Pakete sind in der Paketübersicht erwähnt.

Was tut Samba? Eigentlich nimmt es nur das, was der Server bereits anbietet, und stellt es über die Protokolle SMB/CIFS zur Verfügung. Das bedeutet im Einzelnen:

- Die zu exportierenden Verzeichnisse oder Festplatten müssen in den Dateibaum eingebunden sein,
- die Kennungen, die darauf Zugriff haben sollen, müssen auf dem System existieren, und
- die Drucker, auf denen gedruckt werden soll, müssen im lokalen Drucksystem installiert und konfiguriert sein.

Sie finden die Konfiguration für Samba i. d. R. im Verzeichnis `/etc/samba`. Dort sollte bereits eine Datei `smb.conf` oder eine Datei ähnlichen Namens liegen, die Sie als Vorlage für Ihre Samba-Konfiguration benutzen können. Starten Sie einen Texteditor Ihrer Wahl, und laden Sie die `smb.conf`. Wir werden mit ein paar einfachen Änderungen beginnen, Samba ans Laufen zu bringen.

Tipp

Die SMB/CIFS-Services belegen direkt eine Reihe von Ports, die Sie in der Firewall für den Zugriff von außen öffnen müssen: Die Ports 137, 138 und 139 werden von NetBIOS für verschiedene Dinge wie Auflösung der Windows-Namen etc. genutzt. Die Datenübertragung wird dann über Port 445 abgewickelt.

6.1 Mit Samba eine Arbeitsgruppe bilden

Die Konfigurationsdatei ist über Einträge der Form `[Schlüsselwort]` in Sektionen gegliedert. Im ersten Schritt wollen wir den Teil anpassen, der die globalen Einstellungen enthält. Dieser ist ganz oben zu finden und heißt `[global]`.

```
[global]

# workgroup = NT-Domain-Name or Workgroup-Name, eg: MIDEARTH
   workgroup = ABT2.5.3
```

```
# server string is the equivalent of the NT Description field
   server string = Fileserver Abt. 2.5.3

# Security mode. Defines in which mode Samba will operate. Possible
# values are share, user, server, domain and ads. Most people will want
# user level security. See the Samba-HOWTO-Collection for details.
   security = user
```

Betrachten wir zuerst die Eintragungen `workgroup` und `security`. Mit ihnen legen Sie fest, wie sich Ihr Server im Verbund mit anderen Windows-Rechnern verhält. Wir wollen Ihnen im Folgenden zwei Möglichkeiten der Konfiguration erläutern, eine einfache, in der der Linux-Server einen alleinstehenden Server in einem domain-controller-losen Netzwerk darstellt, und eine komplizierte, in der Samba an einen Domain-Controller angebunden werden muss[1].

Samba, alleinstehend: Windows-Rechner beweisen ihre Zugehörigkeit zueinander, indem sie sich einen Familiennamen bzw. einen Arbeitsgruppennamen geben. Diese etwas anarchische, unregulierte Methode sorgt in kleinen Netzwerken oft für Verwirrung, da bei Stand-alone-PCs bei der Installation selten daran gedacht wird, hier einen vernünftigen, geschweige denn den gleichen Namen zu verwenden. Erst später, wenn man sich der Fähigkeiten des Netzwerks bewusst wird, führt das Nichtauffinden der »benachbarten Computer« dazu, dass der Name der Arbeitsgruppe auf allen Rechnern angepasst wird. Wenn Sie also schon vernetzten Windows-Clients den Zugriff auf Ihren Samba-Server erlauben wollen, sollten Sie hier die Arbeitsgruppe eintragen, denen diese angehören.

Als Sicherheitslevel wählen wir in diesem Fall `security = user`. Dies bedeutet hier, dass für jeden Nutzer, der von einem anderen Windows-Rechner auf Freigaben auf dem Linux-Server zugreifen will, eine Kennung existieren muss, die möglichst gleich der Kennung ist, mit der derjenige sich auf seinem PC unter Windows anmeldet. Außerdem kommen wir nicht darum herum, ihn einmal am Linux-Rechner ein Passwort eingeben zu lassen, was ebenfalls das gleiche sein sollte, wie er es an seinem PC benutzt. Das erspart Ihnen hinterher Supportaufwand und dem Nutzer die lästige doppelte Passworteingabe.

Warum ist das so, und wie muss das Passwort eingegeben werden? Die neueren Windows-Versionen nutzen bei der Übergabe des Passworts eine Verschlüsselung, die nicht mit den unter Linux genutzten Algorithmen kompatibel ist. Daher ist das eigentlich sinnvolle Feature, Passwörter nicht im Klartext über ein Netzwerk zu übertragen, hier eine Quelle von Unannehmlichkeiten: Samba muss eine eigene Passwort-Verwaltung vorhalten, um den Zugriff von Windows-Clients aus zu ermöglichen. Diese kann über das Kommando `smbpasswd` gepflegt werden. Um eine Kennung, die bereits auf dem Server eingerichtet ist, auch unter Samba nutzen zu können, würde man wie folgt vorgehen:

[1] Wie heißt es so schön in vielen Anleitungen und HOWTOs: Wenn Sie nicht wissen, was ein Domain-Controller ist, lesen Sie nur den einfachen Teil. :-)

```
hugo:~# smbpasswd -a victor
New SMB password: # Ein Passwort wird eingegeben
Retype new SMB password: # Es wird nochmal eingegeben
Added user victor.
hugo:~#
hugo:~#
```

Für die root-Kennung sollten Sie ebenfalls mit `smbpasswd` ein Passwort festlegen. Dieses darf sich vom Systempasswort des Rechners unterscheiden – und sollte es aus Sicherheitsgründen auch.

Samba als Mitglied einer Windows-Domäne: Windows-Netzwerke sind eine komplexe Angelegenheit, die von ihren Administratoren oft unterschätzt wird. Die Bereiche der Samba-Dokumentation, die sich mit diesen Problemen beschäftigen und die Archive der Mailingliste sind voll von Beispielen falscher Konzeptionen oder technischer Umsetzungen. Da dies ein Buch über das Netzwerken mit Linux und nicht mit Windows ist, werden wir uns hier auf eine einfache Einbindung eines Samba-Servers in eine bestehende Windows-Domäne beschränken. Sollte der einfache Weg bei Ihnen nicht funktionieren, sollten Sie die Samba-Dokumentation auf Ihrem Server installieren (oder direkt auf `www.samba.org` lesen) und den Hinweisen folgen, die »The Official Samba-3 HOWTO and Reference Guide« in ausreichender Menge enthält[2].

In der Konfigurationsdatei `smb.conf` ändern Sie das Sicherheitsmodell von `user` in `domain`:

```
[global]
   workgroup = ABT2.5.3
   server string = Fileserver Abt. 2.5.3
   security = domain
```

Jetzt müssen Sie im Windows-Domain-Controller einen Computer-Account für Ihren Linux-Server anlegen. Danach können Sie mit dem folgenden Kommando Ihren Linux-Server der Domäne beitreten lassen:

```
hugo:~# net rpc join -U Administrator
Password: # Das Passwort der Administratorkennung des Windows-Servers
Joined ...
hugo:~# /etc/init.d/samba restart
Stopping Samba daemons: nmbd smbd.
Starting Samba daemons: nmbd smbd.
hugo:~#
```

[2] <Werbeblock>Bei Addison-Wesley gibts die gebunden und auf Deutsch! (ISBN 3-8273-2415-7) :-) </Werbeblock>

Je nach auf dem Windows-Server verwendeten System meldet Samba »Joined domain ABT2.5.3« (altes NT4-Domainsystem) oder »Joined hugo to realm ABT2.5.3« (bei Active Directory Service). Danach muss Samba neu gestartet werden.

Tipp

Es müssen trotzdem auf dem Linux-Server die gleichen Kennungen vorhanden sein, die auf dem Windows-Server eingerichtet wurden, und sie müssen gültig sein, d. h. es muss ein Einloggen auch auf anderem Weg, z. B. mit `ssh` für diese Kennungen möglich sein. Es müssen jedoch nicht unbedingt Passwörter für die Kennungen eingetragen sein, da diese ja vom Domain-Controller verwaltet werden.

Existiert die Kennung unter Linux nicht, kann der Nutzer von seinem Windows-Client aus nur die frei zugänglichen Services nutzen. Außerdem können Sie bereits unter Linux eingerichtete Kennungen sperren, wenn Sie in der Datei `/etc/passwd` statt einer normalen Shell `/bin/false` oder einen anderen Wert eintragen, den es nicht als Shell gibt.

6.2 Verzeichnisse freigeben

Im Prinzip funktioniert die Freigabe von Verzeichnissen so, wie man es von Windows kennt, nur mit dem Unterschied, dass die notwendigen Informationen in einer Textdatei eingetragen werden und dies i. d. R. nur dem Administrator (also root) erlaubt ist. Linux als Basis bringt jedoch noch ein paar Besonderheiten mit:

- das sogenannte **Home-Verzeichnis** des Nutzers kann mit einem gesonderten Konfigurationseintrag freigeschaltet werden. Damit bekommt jede Kennung einen eigenen, privaten Bereich. Dabei sieht jeder Nutzer eine Freigabe mit dem Namen `home`, die jedoch bei jeder Kennung auf einen anderen Bereich zeigt.
- Konfigurationen gelten immer für das Verzeichnis und alle Unterverzeichnisse, solange ein Unterverzeichnis nicht mit einer eigenen Sektion in der Konfigurationsdatei erscheint.
- Es existieren verschiedene Optionen, mit denen Sie die unter Linux übliche Rechtevergabe bezogen auf Kennungen und Gruppen auf das freigegebene Verzeichnis umsetzen können.

In der Beispielkonfiguration ist eine ganze Reihe von Einträgen vorhanden, mit denen spezielle Verzeichnisse für die Nutzung im Netzwerk freigegeben werden. Nehmen wir einmal an, wir wollen auf dem Linux-Server

- ein Verzeichnis einrichten, in dem die Nutzer des Netzwerks Backups anlegen können. Die Verzeichnisse sollten gegen das Lesen und Schreiben durch einen der anderen Nutzer geschützt sein, es sollte aber möglich sein, dort Unterverzeichnisse anzulegen.

- Dann brauchen wir ein Verzeichnis, in dem Anwendungssoftware gespeichert werden kann. Dies soll die Installation von Windows-Software auf den Windows-Clients erleichtern.
- Und zum Schluss hat eine Arbeitsgruppe darum gebeten, einen gemeinsamen Bereich für die Mitglieder der Arbeitsgruppe zu bekommen.

Beginnen wir mit dem Home-Verzeichnis: Diese Sektion ist bereits vorgegeben und kann im Prinzip so übernommen werden. Die Optionen `create mask` und `directory mask` sorgen dafür, dass neue Dateien nur mit Schreibberechtigung für den Nutzer angelegt werden. Die Syntax der Zahlenwerte orientiert sich dabei am numerischen Modus des Kommandos `chmod`.

```
[homes]
        comment = Home Directories
        read only = No
        create mask = 0600
        directory mask = 0700
        browseable = No
```

Mit der Option `browseable` konfigurieren Sie im Normalfall, ob eine Freigabe beim »Browsen«, also dem Zugriff über die Netzwerkumgebung, angezeigt wird oder nicht. Bei den Home-Verzeichnissen hat diese Option eine besondere Funktion: Es wird keine Freigabe `homes` angezeigt, das jeweilige Home-Verzeichnis eines Nutzers aber schon.

Als nächstes kümmern wir uns um das Backup-Verzeichnis: Unser Linux-Server hat dafür einen Bereich eines Raid-Arrays im Verzeichnis `/srv/backup` eingebunden. Das Verzeichnis ist bereits eingerichtet; alle Nutzer besitzen dort ein Unterverzeichnis, in das sie schreiben können. Dann können wir über folgende Konfiguration dieses Verzeichnis im Netzwerk zur Verfügung stellen:

```
hugo:/srv# cat >> /etc/samba/smb.conf <<EOF
[Backup]
        comment = Backup-Bereich
        path = /srv/backup
        read only = No
        create mask = 0600
        directory mask = 0700
EOF
# Sie können aber auch einen Texteditor benutzen :-)
hugo:/srv# ll
drwxrwx---   3 root root 1.0K Mar 22 16:07 backup
...
hugo:/srv# ll backup
...
drwx------  3 gutmann root   4.0K Mar 22 15:43 gutmann
drwx------  2 lannert root   4.0K Mar 22 15:46 lannert
hugo:/srv#
```

Die erste Bedingung, dass die Nutzer sich nicht gegenseitig in die Verzeichnisse schauen können, haben wir auf der Dateisystem-Ebene umgesetzt: Dadurch, dass die Verzeichnisse nur für die Kennung lesbar und schreibbar sind, sollte hier außer root keine andere Kennung hineinschauen können. Zusätzliche Sicherheit erreichen wir dadurch, dass wir Samba mit den Optionen `create mask` und `directory mask` beauftragen, neue Dateien und Verzeichnisse ebenfalls nur für den Nutzer les- und schreibbar zu machen. Wenn jetzt der Herr Gutmann über das Netzwerk ein Verzeichnis `Test` anlegt, sieht das Ergebnis im System so aus:

```
hugo:/srv# ll backup/gutmann/
total 4.0K
drwx------  2 gutmann root   4.0K Mar 22 15:43 Test
hugo:/srv#
```

Jetzt geht es um das Verzeichnis mit der Anwendungssoftware: Wir haben zwei Kollegen, die Herren Gutmann und Lannert[3], die in der Lage sein sollen, dort Dateien und Verzeichnisse anzulegen, die dann von den anderen Nutzern des Systems gelesen, also in diesem Fall für die Installation genutzt werden sollen.

```
hugo:/srv# cat >>/etc/samba/smb.conf <<EOF
[Software]
        comment = Software
        path = /srv/win_software
        read only = Yes
        write list = gutmann, lannert
        create mask = 0644
        directory mask = 0755
EOF
hugo:/srv# mkdir win_software
hugo:/srv# chmod a+rwx win_software
# Die Notwendigkeit dieser Rechtevergabe erklären wir weiter unten
hugo:/srv# ll
...
drwxrwxrwx   3 root root 1.0K Mar 22 16:07 win_software
hugo:/srv#
```

Hier hatten wir zusätzlich die Bedingung, dass die Freigabe `Software` für alle Nutzer nur lesbar sein sollte, jedoch die Kennungen `gutmann` und `lannert` dort schreibberechtigt sein sollten. Dies lässt sich durch die Option `read only = Yes` regeln, die durch die in `write list` angegebenen Kennungen eingeschränkt wird. Wenn jetzt einer der Herren Dateien in dieses Verzeichnis kopiert, könnte das Ergebnis so aussehen:

[3] Unsere Beispiele sind halt aus dem Leben gegriffen :-)

```
hugo:/srv# ll win_software
total 28M
-rw-r--r--  1 gutmann gutmann  28M Jun  8  2004 Netscape-7.1-Setup.exe
drwxr-xr-x  2 gutmann gutmann 1.0K Mar 22 16:06 gimp
-rw-r--r--  1 gutmann gutmann 348K Oct 19  2003 putty.exe
hugo:/srv#
```

Das letzte Beispiel hat in dieser Form jedoch einen kleinen Haken: Das Verzeichnis `win_software` muss für alle Nutzer schreibbar gemacht werden, damit die Kennungen `gutmann` und `lannert` dort schreibberechtigt sind. Die `create mask` und die `directory mask` sorgen dafür, dass die in diesem Verzeichnis angelegten Dateien auch auf der Linux-Ebene von keiner anderen Kennung verändert werden können. Aber durch die Nutzung anderer File-Transfer-Dienste, wie z. B. SCP oder FTP wäre es für alle Nutzer möglich, dort Dateien unterzubringen. Außerdem kann die Kennung `lannert` die mit der Kennung `gutmann` angelegten Dateien nicht wieder löschen. Wir sollten das Beispiel daher etwas erweitern: Zuerst sollten Sie mit den Mitteln Ihrer Distribution eine neue Gruppe anlegen. Wir wählen für unser Beispiel den Gruppennamen `winsoft`. Dann sollten Sie die notwendigen Kennungen dieser Gruppe zuordnen. Wir haben das schon getan, wie Sie anhand des `getent`-Kommandos erkennen können:

```
hugo:/srv# getent group winsoft
winsoft:*:907:gutmann,lannert
hugo:/srv# chgrp winsoft win_software
hugo:/srv# chmod g+s,o-w win_software
hugo:/srv# ll
...
drwxrwsr-x   3 root winsoft 1.0K Mar 22 16:07 win_software
hugo:/srv#
```

Jetzt kann auf der Linux-Ebene das Verzeichnis nicht mehr von allen Nutzern beschrieben werden. Außerdem hilft uns das s-Bit dahingehend, dass alle Dateien und Verzeichnisse, die in `win_software` angelegt werden, ebenfalls die Gruppe `winsoft` zugeordnet bekommen. Jetzt muss nur noch die Samba-Konfiguration dementsprechend angepasst werden:

```
[Software]
        comment = Software
        path = /srv/win_software
        read only = Yes
        write list = gutmann, lannert
        create mask = 0664
        directory mask = 0775
```

Die Einträge für das Erzeugen von Dateien und Verzeichnissen wurden so abgeändert, dass auch Samba sie mit Schreibberechtigung für die Gruppe anlegt. Jetzt würden nach einem Kopieren die Dateien folgende Berechtigungen haben:

```
hugo:/srv# ll win_software
total 28M
-rw-rw-r--  1 gutmann winsoft  28M Jun  8  2004 Netscape-7.1-Setup.exe
drwxrwsr-x  2 gutmann winsoft 1.0K Mar 22 16:06 gimp
-rw-rw-r--  1 gutmann winsoft 348K Oct 19  2003 putty.exe
hugo:/srv#
```

Die im letzten Beispiel einzurichtende Arbeitsgruppe unterscheidet von der Konfiguration für das Software-Verzeichnis in der Hauptsache dadurch, dass nicht alle Nutzer dort leseberechtigt sein dürfen. Auf der Linux-Ebene sollten Sie die gleichen Vorrausetzungen schaffen, wie sie für das Software-Verzeichnis notwendig waren: Es sollte eine Gruppe angelegt werden, der die zur Arbeitsgruppe gehörenden Kennungen zugeordnet werden. Dann legen Sie ein Verzeichnis an, dass für die Gruppe les- und schreibbar gemacht wird. Das s-Bit sollten Sie ebenfalls setzen:

```
hugo:/srv/ag# mkdir linuxmigration
hugo:/srv/ag# getent group migration
migration:*:908:gutmann,lannert
hugo:/srv/ag# chgrp migration linuxmigration
hugo:/srv/ag# chmod g+rws,o-rwx linuxmigration
hugo:/srv/ag#
```

Die Liste der erlaubten Nutzer wird diesmal über die Option `valid users` spezifiziert. Mit `public = No` wird der Zugriff »für den Rest der Welt« abgeklemmt. Da jetzt eine Liste von Nutzern existiert, die ausschließlich Zugriff auf diese Freigabe haben, kann mit `writeable = Yes` eine generelle Schreibberechtigung gesetzt werden. Die Masken für die Datei- und Verzeichniserzeugung sorgen dafür, dass die »anderen« auch auf der Linux-Ebene die in diesem Verzeichnis liegenden Daten nicht lesen können:

```
[Migration]
        comment = AG Linux-Migration
        path = /srv/ag/linuxmigration
        public = No
        writeable = Yes
        valid users = +migration
        create mask = 0660
        directory mask = 0770
```

Da wir uns schon die Arbeit machen müssen, die Systemgruppe `migration` zu pflegen, können wir die Samba-Konfiguration in einem Punkt noch vereinfachen: Anstatt bei `valid users` die Kennungen explizit anzugeben, können wir an dieser Stelle auch mit `+migration` direkt einen Verweis auf die Systemgruppe eintragen. Wenn jetzt dort Verzeichnisse oder Dateien abgespeichert werden, sieht das auf der Systemebene folgendermaßen aus:

```
hugo:/srv# ll linuxmigration
insgesamt 136K
drwxrws---  2 gutmann migration 4,0K 2005-08-29 21:33 Eigene Dateien
-rw-rw----  1 gutmann migration 125K 2005-08-29 21:32 wordless.doc
hugo:/srv#
```

Vielleicht noch ein letzter Hinweis: Mit der Option `browseable = no` schalten Sie die Funktion ab, dass die Freigabe beim »Browsen«, d. h. in der Netzwerkumgebung in der Liste der Freigaben des ausgewählten Rechners angezeigt wird. In diesem Fall kann man auf die Freigabe nur mit dem kompletten Pfad `//hugo/Migration` zugreifen.

6.3 Drucken mit Samba

Wie oben bereits erwähnt, ist Samba kein eigenes Drucksystem, sondern baut auf bereits eingerichteten Drucksystemen auf. Verwenden Sie z. B. CUPS, werden alle Drucker, die im System eingerichtet sind, durch folgende Konfiguration eingebunden:

Listing 6.1: smb.conf

```
[global]
...
printing = cups
...
[printers]
        comment = Alle Drucker der Abt. 2.5.3
        printable = Yes
        writable = no
        guest ok = no
        path = /var/tmp
        create mask = 0700
```

Folgende Informationen sind notwendig: In der globalen Konfiguration wird als Drucksystem `cups` eingetragen. Die Sektion `printers` ist neben `global`, `home` und `print$` einer der Standardkonfigurationsbereiche. Diese Konfiguration erlaubt es allen Nutzern, die eine Kennung auf diesem Rechner besitzen, die über Samba exportierten Drucker zu nutzen. Wenn Sie beim Drucken keine Einschränkungen machen wollen, können Sie die Option `public = Yes` hinzufügen. Die Angaben `path` und `create mask` beziehen sich nur auf das Verzeichnis, in dem die Druckdateien bis zur Fertigstellung des Drucks zwischengespeichert werden. Daher ist hier auch mit `/var/tmp` ein Temporärverzeichnis vorgegeben.

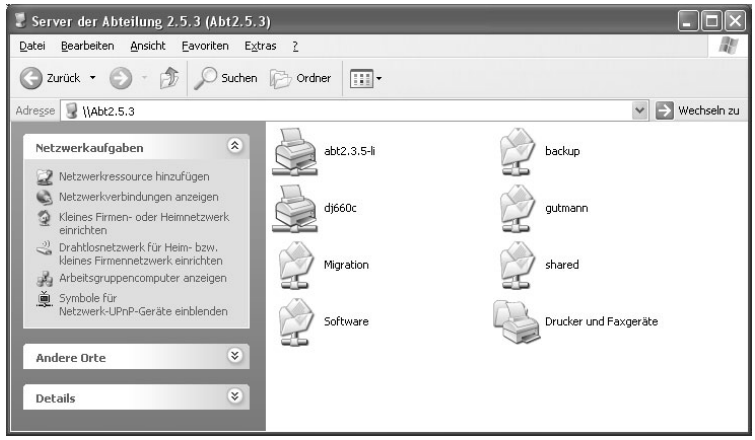
Leider ist es damit noch nicht getan: um Windows-Clients jetzt das Drucken zu ermöglichen, müssen noch die passenden Druckertreiber installiert werden. Windows-Clients erwarten, dass die Druckerfreigabe auf einem Netzwerkrechner nicht nur die Möglichkeit zu drucken sondern auch die dazu gehörigen Treiber anbietet.

Diese müssen daher für alle Windows-Versionen installiert werden, die auf Clients installiert sind, die diesen Drucker über das Netz benutzen wollen. Samba stellt dafür die spezielle Freigabe `print$` zur Verfügung:

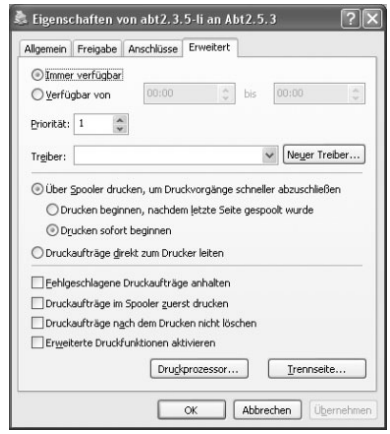
```
[print$]
        comment = Druckertreiber
        path = /var/lib/samba/printers
        admin users = root, gutmann, lannert
        write list = root, gutmann, lannert
```

Hier sind außer der Angabe von Kennungen, die in diesem Bereich schreibberechtigt sind, keine besonderen Angaben notwendig. Samba weiß, wie es die über diese Freigabe zur Verfügung gestellten Daten behandeln muss. Wir haben uns die Arbeit hier einfach gemacht und die berechtigten Kennung nicht nur in die `write list` eingetragen, sondern sie auch zu `admin users` gemacht. Diese Option sollte mit Vorsicht angewandt werden, da alle Aktionen, die von diesen Nutzern ausgeführt werden, auf der Linux-Systemebene effektiv mit root-Berechtigung ablaufen. D. h. z. B., dass alle Daten, die dort gespeichert werden, der Kennung root gehören. Da Samba hier aber die Kontrolle über angelegte Dateien und Verzeichnisse übernimmt, ist das Riskiko an dieser Stelle vernachlässigbar.

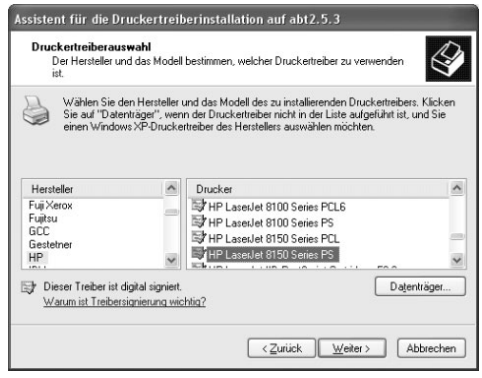
Wie können Sie jetzt Druckertreiber für die auf dem System bekannten Drucker einrichten? Diese Aufgabe erledigen Sie mit dem Mitteln von Windows! Dazu wechseln wir jetzt kurz das Betriebssystem: Wenn Sie unter Windows über die Netzwerkumgebung sich die Freigaben des Rechners `abt2.5.3` ansehen, finden Sie dort neben den Druckern und den Datenbereichen ein Icon mit dem Namen DRUCKER UND FAXGERÄTE.



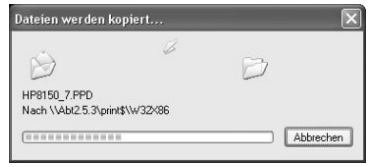
Dahinter verbirgt sich der Bereich, der über die Konfiguration `print$` freigegeben wurde. Dort finden Sie wiederum alle auf dem Samba-Server installierten Drucker. Jetzt können Sie über das Kontextmenü des Druckers den Dialog EINSTELLUNGEN aufrufen und den Reiter ERWEITERT auswählen.



Das Listenfeld TREIBER zeigt Ihnen die Druckertreiber an, die der Samba-Server schon kennt. Darüber könnten Sie dem Drucker jetzt einen bereits installierten Treiber zuweisen. Fehlt der passende Treiber, starten Sie über den Button NEUER TREIBER einen Wizard, der Sie durch die Installation eines Druckertreibers führt.



Hier stehen Ihnen alle Treiber zur Verfügung, die Windows kennt. Oder Sie wählen einen Treiber von einem DATENTRÄGER. Wenn Sie Glück haben und Rechner und Netzwerkverbindung langsam genug sind, können Sie noch einen Blick darauf erhaschen, wohin Windows die Dateien des ausgewählten Druckertreibers installiert:



Vergessen Sie nicht, den Dialog mit OK zu beenden. Wenn Sie den Drucker vorher ausprobieren, bekommen Sie eine Fehlermeldung!

Danach können Sie den Netzwerkdrucker aufrufen; es werden zuerst die Treiber installiert, danach öffnet sich das Druckerwarteschlangenfenster. Ein Blick in das Verzeichnis, dass wir oben als Speicherplatz für die Druckertreiber angegeben haben, zeigt die Dateien, die installiert worden sind:

```
hugo:/var/lib/samba/printers# ll W32X86
total 4.0K
drwxr-xr-x  2 root gutmann 4.0K Mar 23 10:20 3
hugo:/var/lib/samba/printers# ll W32X86/3
total 5.8M
-rwxr--r--  1 root gutmann  78K Jul 21  2001 HP8150_7.ppd
-rwxr--r--  1 root gutmann  10K Aug  4  2004 HPCJRRPS.DLL
-rwxr--r--  1 root gutmann  23K Aug 18  2001 HPCJRUI.DLL
-rwxr--r--  1 root gutmann  34K Jul 28  2001 HPCLJX.HLP
-rwxr--r--  1 root gutmann  22K Jul 21  2001 HPDJ520.GPD
-rwxr--r--  1 root gutmann 130K Aug 18  2001 HPDJRES.DLL
-rwxr--r--  1 root gutmann 131K Aug  4  2004 PS5UI.DLL
-rwxr--r--  1 root gutmann  25K Jul 28  2001 PSCRIPT.HLP
-rwxr--r--  1 root gutmann 775K Jul 17  2004 PSCRIPT.NTF
-rwxr--r--  1 root gutmann 454K Aug  4  2004 PSCRIPT5.DLL
-rwxr--r--  1 root gutmann  17K Aug 18  2001 hpcabout.dll
-rwxr--r--  1 root gutmann 9.0K Aug 18  2001 hpcstr.dll
-rwxr--r--  1 root gutmann  185 Jul 21  2001 hpljps1.ini
hugo:/var/lib/samba/printers#
```

Wenn Sie die für die Installation eines Druckertreibers notwendigen Dateien kennen, können Sie diese auch »zu Fuß« dort unterbringen, ohne dafür einen Windows-Rechner bemühen zu müssen. Wenn Sie das `printers`-Verzeichnis in eine andere Samba-Installation auf einem anderen Rechner kopieren, stehen Ihnen die Druckertreiber dort ebenfalls zur Verfügung.

6.4 Konfiguration über einen Webbrowser mit SWAT

Auch wenn die Konfigurationsdatei von Samba schon ein Muster an Übersichtlichkeit ist, kann es u. U. angenehmer sein, Änderungen an der `smb.conf` mit Menüführung und direkten Links auf die Hilfe-Seiten vorzunehmen. Außerdem kann über SWAT auch normalen Nutzern die Möglichkeit gegeben werden, online ihr Passwort zu ändern, was die Umständlichkeit der Samba-eigenen Passwort-Verwaltung stark reduziert.

SWAT bringt außer den in der `smb.conf` deklarierten Nutzungsrechten selbst keine Zugriffsbeschränkungen mit. Daher sollte man es als Service des Inetd-Dämons starten. SWAT besitzt auch keine eigene Möglichkeit eines mit SSL verschlüsselten Zugriffs. Daher sollten Sie sich überlegen, Tools wie `stunnel` für den sicheren Zugriff zu nutzen, wenn Sie SWAT häufiger einsetzen wollen.

SWAT über den Inetd-Dämon starten: Die Datei `/etc/inetd.conf` enthält die Services, die der Inetd-Dämon auf Anforderung startet. Bei der Installation der Samba-Pakete wird i. d. R. bereits ein Eintrag in der Konfigurationsdatei ergänzt, den Sie nur einkommentieren müssen, um SWAT nutzen zu können:

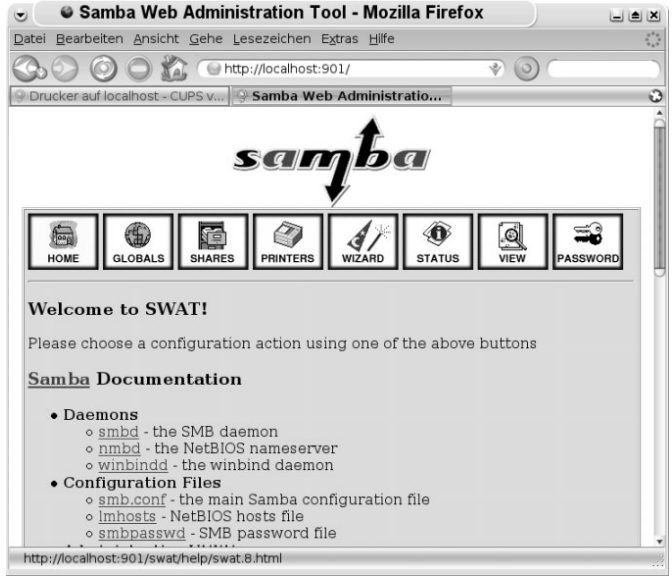
Listing 6.2: /etc/inetd.conf

```
#:OTHER: Other services
# Diese Zeile könnten Sie in Ihrer inetd.conf finden
#<off># swat    stream  tcp  nowait.400   root  /usr/sbin/tcpd  /usr/sbin/swat
# Und so sollte es aussehen, wenn swat gestartet werden soll
swat    stream  tcp  nowait.400   root  /usr/sbin/tcpd  /usr/sbin/swat
```

SWAT benutzt standardmäßig den Port 901, der aber auch in der Datei `/etc/services` eingetragen ist:

```
hugo:~# fgrep swat /etc/services
swat            901/tcp                         # swat
hugo:~# /etc/init.d/inetd reload
Reloading internet superserver: inetd.
hugo:~#
```

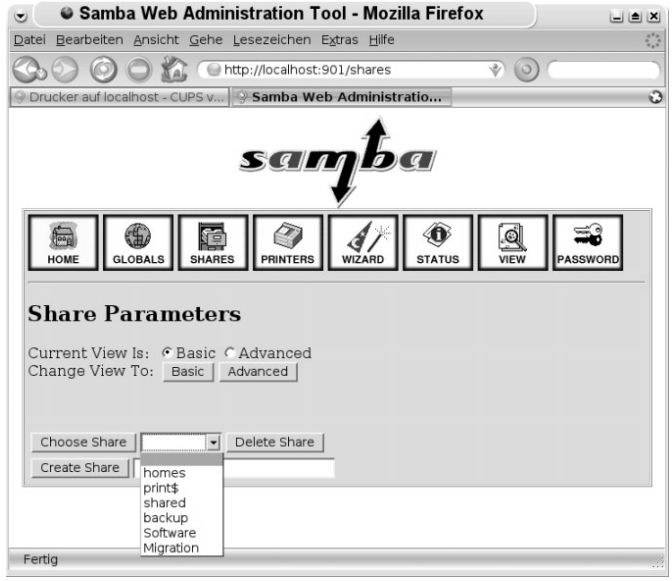
Über den Inetd-Dämon lässt sich der Zugriff auf SWAT über die Dateien `/etc/hosts.allow` und `/etc/hosts.deny` konfigurieren. Wenn Sie in der `hosts.deny` ein `ALL:ALL` stehen haben, sollten Sie zumindest `swat: 127.0.0.1` bzw. die IP-Nummer des Rechners, an dem Sie gerade sitzen, in der `hosts.allow` ergänzen, um über die Adresse `http://localhost:901/` auf SWAT zugreifen zu können.



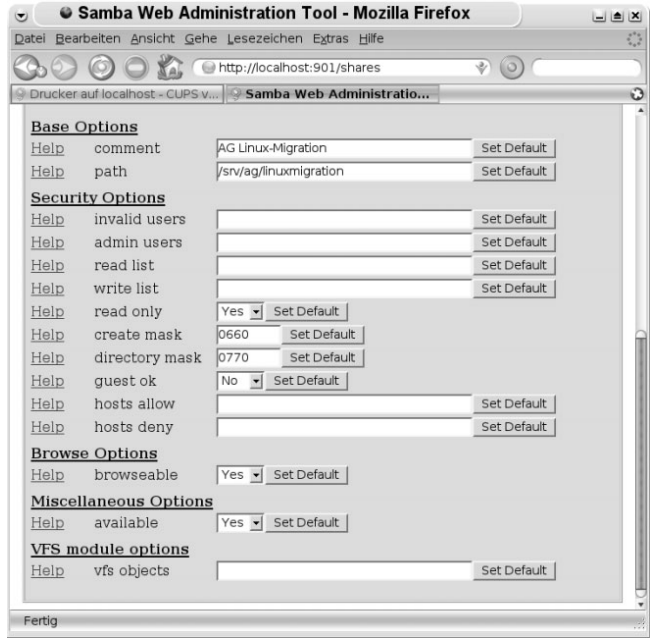
Achtung

Haben Sie bereits eine handgemachte `smb.conf`? So richtig schön mit Kommentaren und Hinweisen für den Bearbeiter und auskommentierten Konfigurationsteilen, die eventuell wiederverwendet werden sollen? In diesem Fall sollten Sie sich eine Sicherheitskopie der `smb.conf` anlegen, da SWAT diese Datei auf das notwendige Minimum reduziert; es werden alle Kommentare entfernt, übrig bleiben nur die aktiven Einträge.

Mit der Adresse `http://localhost:901/` bzw., wenn Sie nicht auf dem Server selbst arbeiten, über die passende IP-Nummer, z. B. `http://192.168.1.10:901/`, steuern Sie SWAT an. Wenn Sie sich mit Administrator-Berechtigung anmelden – d. h., Sie benutzen entweder die Kennung `root` oder haben in der Sektion `[globals]` die Kennung, mit der Sie sich jetzt anmelden, als `admin users` eingetragen –, werden Ihnen über Icons die verschiedenen Konfigurationsmöglichkeiten angeboten. Wir haben einmal die Konfiguration für die Verzeichnisfreigaben über den Button SHARES aufgerufen.

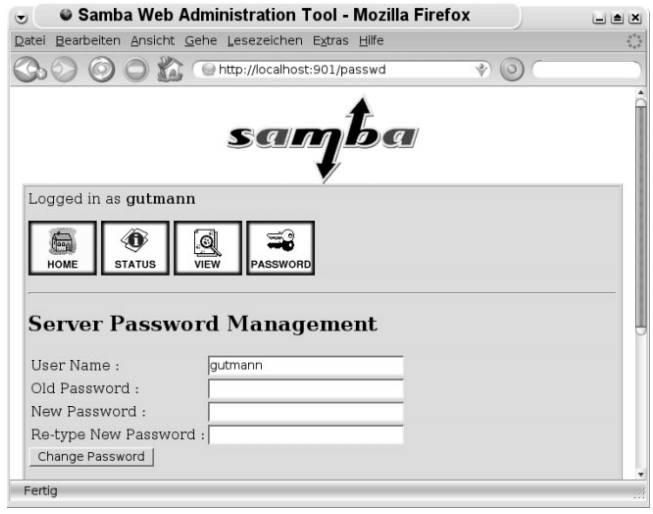


In der Auswahlliste der Freigaben finden Sie alle Einträge, die Sie bereits in der `smb.conf` eingetragen haben. Im Eingabefeld neben CREATE SHARE können Sie den Namen einer neuen Freigabe angeben: Danach gelangen Sie zu einem Eingabeformular, in dem Sie die restlichen Informationen eintragen können. Wir haben einmal die Freigabe für unsere Arbeitsgruppe »Migration« aufgerufen:



Sie können in den Eingabefeldern wiedererkennen, welche Informationen Sie vorher eingegeben haben. Sie werden aber auch feststellen, dass nicht für alle Optionen Eingabefelder existieren. In besonderen Fällen kommen Sie also um eine Bearbeitung der `smb.conf` nicht herum.

Ein Hinweis noch zum Schluss: Nicht-privilegierte Nutzer, die SWAT nutzen, können dort auch nur die Änderungen vornehmen, die ihnen erlaubt sind. Das beinhaltet i. d. R. nur die Änderung des Passworts. Die anderen Konfigurationsoptionen werden dabei gar nicht erst angeboten:



Hinweis: Wenn's mal nicht so geklappt hat ...

Problem: Sie können sich mit der Kennung `root` bei SWAT nicht anmelden.

Analyse: SWAT nutzt für die Zugriffskontrolle die Kennungen aus der internen Passwort-Datenbank. Haben Sie die Kennung `root` dort nicht explizit hinzugefügt, werden Sie sich auch online nicht damit anmelden können.

Lösung: Fügen Sie die Kennung `root` über das Kommando `smbpasswd -a root` der internen Passwort-Datenbank hinzu.

Weitere Literatur

- »Samba 3 – das offizielle Handbuch« von John H. Terpstra, Jelmer Vernooij, Addison Wesley, ISBN: 3-8273-2415-7
- Offizielle Webseite des Projekts Samba: `www.samba.org`
- »Samba-HOWTO-Collection«: Die HOWTO-Collection enthält das eigentliche Handbuch mit vielen Anwendungsbeispielen.
- `man smb.conf`: Referenz für die Optionen der Datei `smb.conf`
- `man chmod`: Für die Syntax der Angabe der Berechtigungen bei den Optionen `create mask` und `directory mask`

6.5 Mit Linux-Clients am Windows-Netzwerk

Die Datei-Manager der großen Desktop-Systeme KDE und Gnome besitzen einfache Zugriffsmöglichkeiten auf Windows-Freigaben. Mit einer speziellen Protokoll-Angabe können Sie z. B. im Konqueror auf einen Windows-Freigabe zugreifen: um auf die Freigabe `migration` auf dem Server `hugo` zuzugreifen, geben Sie dort die Adresse `smb://hugo/migration` ein. Es werden dann Kennung und Passwort abgefragt und Ihnen nach erfolgreichem Einloggen der Blick auf die dort liegenden Dateien und Verzeichnisse gewährt.

Es gibt aber noch weitere Möglichkeiten, auf Datenbereiche zuzugreifen bzw. diese lokal auf dem Linux-Rechner einzubinden. Dazu benötigen Sie entweder die IP-Adresse des Rechners, dessen Freigabe Sie einbinden wollen, oder dessen NetBIOS-Namen.

Achtung

Die NetBIOS-Namen sind von den DNS-Namen unabhängig. Es handelt sich hierbei um ein separates System der Namensvergabe, für dessen Nutzung Samba spezielle Tools bereitstellt. Seit der Verknüpfung von NetBIOS und TCP/IP wird das Protokoll in der Literatur auch mit dem Namen NetBT bezeichnet.

Aufgrund der Art und Weise, wie NetBIOS die Rechnernamen verwaltet, ist die Ermittlung ein wenig schwierig. Konqueror oder Nautilus erleichtern Ihnen diese Aufgabe, da sie die verschiedenen Schritte beim Aufrufen der Adresse `smb://` bereits durchführen. Wenn Ihnen auf dem Rechner, auf dem Sie eine Windows-Freigabe einbinden wollen, diese Programme nicht zur Verfügung stehen, können Sie die im Netzwerk befindlichen Windows-Rechner auch mit Kommandozeilen-Tools ermitteln, die bei jeder Samba-Installation dabei sind.

Übung: NetBT-Namen mit `nmblookup` und `smbclient` ermitteln

Bevor wir den etwas umständlichen Weg gehen, mit Hilfe der Kommandos `nmblookup` und `smbclient` die im Netzwerk befindlichen Rechner zu ermitteln, müssen wir Ihnen einen kurzen Einblick in die Namensverwaltung von NetBT geben.

NetBT funktioniert auf der Basis von Broadcasts, die i. d. R. nicht über einen Router hinaus verbreitet werden. Daher stehen die Namen der Windows-Rechner meistens nur im LAN zur Verfügung. Meldet sich nun ein Rechner im Netzwerk an, sucht er sich denjenigen Rechner in seiner Netzwerkumgebung, der mit der Funktion »Master Browser« konfiguriert worden ist. Dort meldet der neue Rechner den in seiner Konfiguration eingetragenen Namen an. Der Master Browser führt eine Liste über alle Anmeldungen, die in regelmäßigen Zeitabständen von den Clients aktualisiert werden muss. Meldet sich einer der Client-Rechner ab oder hört der Master Browser über einen definierten Zeitraum nichts mehr von diesem, wird der zum Client gehörige Eintrag in der Liste als »nicht mehr benutzt« markiert.

Welche Auswirkungen hat das auf unsere Suche nach den im Netzwerk befindlichen Rechnern? Wir müssen im Prinzip zwei Schritte durchführen: Zuerst müssen wir den für unser Netzwerk zuständigen Master Browser ermitteln. Danach können wir uns dann von diesem die aktuelle Liste der ihm bekannten Clients ausgeben lassen.

Die Ermittlung des NetBT-Namens des Browsers erfolgt dabei in zwei Schritten: Das Kommando `nmblookup` ist eigentlich dafür da, die zu einem Namen gehörende IP-Nummer zu ermitteln. Wir können mit ihm jedoch auch den Master Browser enttarnen. Die dazugehörige Option ist ein wenig kryptisch, aber das soll an dieser Stelle nicht stören:

```
victor@hugo:~$ nmblookup -M -- -
querying __MSBROWSE__ on 192.168.1.255
192.168.1.25 __MSBROWSE__<01>
victor@hugo:~$
```

Damit haben wir die IP-Nummer unseres Windows-Rechners, der den Master Browser mimt. Jetzt können wir mit einem weiteren Aufruf den Namen des Rechners ermitteln; wir machen uns dabei die

Option `-S` zunutze, die eine Reihe von Statusinformationen über den angegebenen Rechner ausgibt:

```
victor@hugo:~$ nmblookup -S -A 192.168.1.25
Looking up status of 192.168.1.25
        Faktotum       <00> -         B <ACTIVE>
        ABT2.5.3       <00> - <GROUP> B <ACTIVE>
        Faktotum       <20> -         B <ACTIVE>
        ABT2.5.3       <1e> - <GROUP> B <ACTIVE>
        ABT2.5.3       <1d> -         B <ACTIVE>
        ..__MSBROWSE__. <01> - <GROUP> B <ACTIVE>

        MAC Address = 00-80-AD-85-6A-2F

victor@hugo:~$
```

Wir wollen hier auf eine genauere Beschreibung dieser Ausgabe verzichten. Uns interessiert an dieser Stelle die erste Zeile, in der der NetBT-Name des Rechners mit der IP-Nummer 192.168.1.25 aufgelistet ist: Faktotum. Jetzt können wir mit dem Kommando `smbclient` die Freigaben des Master Browsers auflisten lassen. Netterweise zeigt er uns dabei auch die ihm bekannten NetBT-Rechnernamen an:

```
victor@hugo:~$ smbclient -N -L Faktotum
Domain=[Faktotum] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]

        Sharename       Type      Comment
        ---------       ----      -------
        IPC$            IPC       Remote-IPC
        ADMIN$          Disk      Remoteadmin
        C$              Disk      Standardfreigabe
Domain=[ABT2.5.3] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]

        Server               Comment
        ---------            -------
        hugo                 hugo (Samba 3.0.10-Debian)
        theodor              theodor (Samba 3.0.10-Debian)

        Workgroup            Master
        ---------            -------
        ABT2.5.3             Faktotum
victor@hugo:~$
```

Unter der Überschrift `Server` finden Sie jetzt die aktuelle Liste der dem Master Browser bekannten Rechner. Jetzt können wir daran gehen, die Freigaben der dort aufgelisteten Rechner wie z. B. `theodor` zu ermitteln und lokal auf unserem Rechner einzubinden.

PS.: Der oben demonstrierte Zugriff auf den Windows-Server geht davon aus, dass ein anonymer Zugriff erlaubt ist. Sollte die Konfiguration des Servers dies verbieten, können Sie mit der Option `-U <Name>` eine Kennung angeben und werden dann nach deren Passwort gefragt. Danach werden Ihnen dann die oben gezeigten Daten mit ähnlichen Inhalten ausgegeben.

Das kommandozeilenorientierte Programm `smbclient`, mit dem die Funktionsfähigkeit eines Zugriffs auf Windows-Freigaben getestet werden kann, haben wir im obigen Workshop bereits gezeigt. Wir wollen kurz ein wenig näher seine Funktionen eingehen:

Dateien kopieren: Wollen Sie ohne großen Aufwand eine Datei von einem Rechner, auf dem keine grafische Oberfläche zur Verfügung steht, auf einen Windows-Rechner oder von dort kopieren, können Sie `smbclient` auch in der Art eines einfachen FTP-Clients benutzen. Sie sollten dazu auf dem Windows-Rechner eine Kennung und Zugriffsrechte auf die gewünschte Freigabe besitzen.

Nehmen wir als Beispiel einmal an, wir wollten eine Datei kopieren und eine Datei drucken:

```
victor@hugo:/tmp$ ll
insgesamt 6M
-rw-r--r--  1 victor victor 5,8M 2005-02-07 21:17 adobe-ps-driver-winstger.exe
-rw-r--r--  1 victor victor 208K 2005-02-21 21:32 test.ps
victor@hugo:/tmp$
```

Der Aufruf von `smbclient` gestaltet sich so: Weil unter Windows Server und deren Freigaben mit einem Backslash (\) getrennt werden, diese aber auf einer Standard-Unix-Kommandozeile eine eigene Bedeutung besitzen, müssen Sie stattdessen Schrägstriche (/) benutzen – die Windows mittlerweile auch versteht. Der Aufruf geschieht dann über die Angabe des Servers und die Freigabe in der Form `//Server/Freigabe`. Wenn Sie nicht mit der Option `-U` eine andere Kennung angeben, wird die gleiche Kennung benutzt, unter der Sie angemeldet sind. Diese muss dann natürlich auch auf dem Windows-Server existieren.

```
victor@hugo:/tmp$ smbclient //Faktotum/Shared
Password:
Domain=[ABT2.5.3] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> help
?              altname        archive        blocksize      cancel
case_sensitive cd             chmod          chown          del
dir            du             exit           get            getfacl
hardlink       help           history        lcd            link
lowercase      ls             mask           md             mget
mkdir          more           mput           newer          open
```

```
print          printmode      prompt         put            pwd
q              queue          quit           rd             recurse
reget          rename         reput          rm             rmdir
setmode        stat           symlink        tar            tarmode
translate      vuid           logon          listconnect    showconnect
!
victor@hugo:/tmp$
```

Der `smbclient` hat eine rudimentäre Online-Hilfe, die zumindest einen Überblick über die zur Verfügung stehenden Kommandos gibt und auf Anforderung deren Syntax beschreibt. Wir haben auf unserem Linux-Server einen Druckertreiber für die Postscript-Ausgabe liegen (`adobe-ps-driver-winstger.exe`), die wir auf dem Windows-Server ablegen wollen. Mit `put` wird die Datei kopiert.

```
smb: \> ? put
HELP put:
       <local name> [remote name] put a file

smb: \> put adobe-ps-driver-winstger.exe
smb: \> dir
  .                                   D        0  Wed Apr 27 11:31:10 2005
  ..                                  D        0  Sun Apr 17 18:22:08 2005
  adope-ps-driver-winstger.exe          5992297  Mon May  7 20:35:29 2005
smb: \> q
victor@hugo:/tmp$
```

Mit `get` würde eine Datei in das Verzeichnis heruntergeladen, aus dem wir den `smbclient` gestartet haben. Weitere wichtige Kommandos sind `mput` und `mget`, mit denen wir mehrere Dateien kopieren können. Aber wenn Sie einmal einen FTP-Client über eine Kommandozeile bedient haben, sollten Ihnen diese Kommandos bekannt vorkommen. Einen etwas ausführlicheren Überblick über die Kommandos und darüber, wie man sie benutzt, finden Sie auf der Manpage (`man smbclient`).

Drucken mit dem `smbclient`: Eine Möglichkeit, die man in diesem Zusammenhang eher nicht erwartet, ist die Druckfunktion, die `smbclient` mitbringt. Sie können sich mit einer Druckerfreigabe verbinden und dann Druckdateien von Ihrem Linux-Rechner auf den Windows-Server kopieren:

```
victor@hugo:/tmp$ ll
insgesamt 6M
-rw-r--r--  1 victor victor 5,8M 2005-02-07 21:17 adobe-ps-driver-winstger.exe
-rw-r--r--  1 victor victor 208K 2005-02-21 21:32 test.ps
victor@hugo:/tmp$ smbclient //Faktotum/abt2.3.5-re
Password:
Domain=[ABT2.5.3] OS=[Windows 5.0] Server=[Windows 2000 LAN Manager]
smb: \> print test.ps
```

```
putting file test.ps as test.ps (683.6 kb/s) (average 683.6 kb/s)
smb: \> put test.ps
putting file test.ps as \test.ps (1574.3 kb/s) (average 953.2 kb/s)
smb: \> q
victor@hugo:/tmp$
```

In diesem Fall sorgen sowohl `put`, als auch das `print`-Kommando dafür, dass die ausgewählte Datei auf dem Windows-Server gedruckt wird. Denken Sie jedoch daran, dass die Datei schon für den Drucker aufbereitet sein, also eine »Druckdatei« sein muss. In unserem Beispiel haben wir eine Postscript-Datei gedruckt. Handelt es sich bei dem Zielgerät nicht um einen Postscript-fähigen Drucker, werden Sie jetzt eine ganze Menge Text auf dem Papier haben …

Freigaben in lokale Verzeichnisse einbinden: Eine weitere nützliche Funktion ist das Einbinden von Windows-Freigaben in lokale Verzeichnisse, ähnlich einem NFS-Mount. Die aktuellen Samba-Versionen unterstützen dabei sogar den User-Mount, d. h. die Einbindung solcher Freigaben ohne root-Berechtigung. Wir gehen wieder davon aus, dass auf dem Server die gleiche Kennung (`victor`) existiert, unter der wir unter Linux angemeldet sind. Sollten Sie sich mit einer anderen Kennung auf dem Windows-Rechner anmelden wollen, können Sie diese mit der Option `-U <Name>` angeben.

```
victor@hugo:~$ mkdir WinShare
victor@hugo:~$ smbmount //Faktotum/Shared WinShare
Password:
victor@hugo:~$ ll WinShare
insgesamt 5,8M
-rw-r--r--  1 victor victor 5,8M 2005-02-07 21:17 adobe-ps-driver-winstger.exe
victor@hugo:~$
```

Auch wenn Konqueror und Nautilus die Möglichkeit bieten, eine Datei aus einer Windows-Freigabe per Click zu bearbeiten, ist es unter Umständen sinnvoller, das Verzeichnis mit den Dateien, die bearbeitet werden sollen, als Unterverzeichnis einzubinden. Die Datei-Manager haben in einigen Fällen Probleme mit den Sperrmechanismen des Windows-Servers, falls eine Datei mehrfach geöffnet wird. Dies wird bei der Einbindung durch `smbmount` berücksichtigt.

Achtung

Das Mounten von Windows- oder Samba-Shares unter einer nichtprivilegierten Kennung funktioniert nur, wenn das Programm `smbmnt` das `suid`-Bit gesetzt hat und dem Nutzer `root` gehört!

Die Einbindung von Windows-Freigaben mit `smbmount` kann auch automatisiert werden. Dazu ist es notwendig, die Kennung und das Passwort, mit der die Anmeldung stattfinden soll, in einer Datei abzulegen. Diese »Credentials« können Sie dann mit der Option `-o credentials=<dateiname>` dem `smbmount`-Kommando mitgeben, sodass z. B. beim Anmelden in der Datei `.profile` die Freigabe eingebunden und in der Datei `.logout` der Befehl zum Unmounten gegeben wird. Ach, den haben wir Ihnen ja bisher unterschlagen: Aber Sie wären bestimmt auch selber darauf gekommen: Mit `smbumount WinShare` können Sie die Verbindung zum Windows-Server wieder lösen.

6.6 Samba wann und wie einsetzen

Samba bringt von seiner Konzeption her Parameter mit, die bei einem Einsatz insbesondere unter Sicherheitsaspekten berücksichtigt werden müssen:

- Die Datenübertragung mit Ausnahme der Passwörter ist unverschlüsselt.
- Die Kommunikation innerhalb des Netzwerks funktioniert über Broadcasts.
- Das Domänen-System wird, weil es einfach zu umgehen ist, mit steigender Anzahl von Clients unübersichtlich.

Die fehlende Verschlüsselung bei der Datenübertragung stellt uns vor wichtige Entscheidungen, wenn wir ein sicheres Netzwerk einrichten wollen:

- Trauen wir unserem Netzwerk?
- Wie weit lassen wir den Datenaustausch zu?
- Wie viel Freiheit lassen wir bei der Nutzung des Netzwerks zu?

Ein sternförmiges, mit Hilfe von Switches aufgebautes Netzwerk bietet eine relative Sicherheit gegen das Abhören einzelner Verbindungen. Diese sind nämlich – wenn der Switch richtig konfiguriert ist – Punkt-zu-Punkt-Verbindungen zwischen Server und Client, zwischen die sich i. d. R. kein anderer Rechner pfuschen kann. Aber schon bei der Verwendung von Hubs kann es dazu kommen, dass Pakete einer Verbindung von einem dritten Rechner mitgehört werden können. Es mag jetzt nach beginnender Paranoia aussehen, hier den Kolleginnen und Kollegen einen bösen Willen zu unterstellen; Untersuchungen über Computerkriminalität in Unternehmen zeigen jedoch, dass die Zahl derjenigen Delikte deutlich zunimmt, die einen Diebstahl von sensiblen Daten durch entlassene oder unzufriedene Mitarbeiter beinhalten. Eine Kontrolle darüber, welche Rechner sich wie in das Netzwerk einhängen können, erhöht daher die Sicherheit Ihres Samba-Netzwerkes. Sicherheit ist jedoch immer eine Frage der Abwägung: Wenn der Chef seinen Laptop ins Netzwerk einstöpseln und dann Zugriff auf alle Daten haben möchte, müssen Kompromisse zwischen Sicherheit und Einfachheit der Nutzung gemacht werden. Wichtig ist an dieser Stelle nicht, auf Kompromisse zu verzichten, sondern sich darüber im Klaren zu sein, dass man Kompromisse macht und welche Auswirkungen diese haben.

Da Samba den CIFS-Request über das IP-Protokoll abwickelt, kann im Prinzip auch über externe Internet-Verbindungen auf einen Samba-Server zugegriffen werden.

Man sollte sich jedoch gut überlegen, ob man lokale Samba-Server aus Gründen der Bequemlichkeit für den weltweiten Zugriff öffnet, nur weil der Chef von zu Hause aus an seine Dokumente herankommen möchte. Eine Lösung für dieses Problem wäre eine Bereitstellung des Datenbereichs über eine SSL-verschlüsselte WebDAV-Verbindung mit Hilfe des Apache-Servers (siehe Abschnitt 9.7). Eine andere Lösung wäre die Einwahl über ein VPN (siehe Abschnitt 11.2). VPNs sind ebenfalls gut geeignet, geografisch verteilte lokale Netze über eine Internetverbindung zu einem virtuellen, lokalen Netzwerk zu verbinden.

Wichtige Funktionen des Protokolls SMB/CIFS sind mit Hilfe von Broadcasts realisiert. Das hat zur Folge, dass mit jedem ins Netzwerk eingebundenen Client die Netzlast durch den protokollinternen, organisatorischen Datenaustausch steigt. Diese Broadcasts müssen unter umständen auch in alle Subnetze geroutet werden, wodurch sich die Belastung im gesamten lokalen Netz steigt und an Flaschenhälsen wie z. B. einer VPN-Anbindung über eine DSL-Verbindung schnell zu Verstopfungen führen können. Hier hilft es, vorher das Design der Verknüpfung der verschiedenen, lokalen Netzwerke so zu planen, dass Flaschenhälse und unnötiger Netzverkehr vermieden wird.

Fehlkonfigurationen sind häufige Fehlerursachen in einem SMB/CIFS-Netzwerk. Das Märchen von der Einfachheit einer Netzwerkeinrichtung und der Intuitivität einer grafischen Oberfläche verführt viele Computernutzer dazu, Dinge in einem Netzwerk zu tun, die i. d. R. das Ego des Nutzers, aber nicht die Funktionsfähigkeit des Netzwerks erhöhen. Hier verlaufen die Fronten des ewigen Kampfes zwischen Administrator und Anwender; um diesen Kampf zu vermeiden ist eine Strategie notwendig, die mit möglichst wenigen Restriktionen auskommt, aber viel Komfort bietet. Sich als Administrator auf ausgewählte technischen Gegebenheiten zurückzuziehen und mit diesen Restriktionen zu begründen, hat maßgeblich zum Misstrauen der Anwender gegenüber ihren EDV-Verantwortlichen beigtragen. Eine Laissez-Faire-Strategie ist aber ebenfalls keine Lösung und wird mit größter Wahrscheinlichkeit dazu führen, dass Sie mehr auf Viren- und Trojaner-Jagd sind, als sich um den Support für Ihre Anwender zu kümmern.

Achten Sie daher bei der Einrichtung Ihres Samba-Netzwerks darauf,

- die einzelnen Subnetze so einzurichten, dass ein falsch konfigurierter Rechner nicht das gesamte Netzwerk lahmlegt,
- dass Sie den Überblick behalten können, wo wie viele und was für welche Rechner im Netzwerk eingebunden sind und welche Personen daran arbeiten,
- dass Ihre Anwender das Netzwerk noch verstehen und die Einschränkungen, die Sie vorgeben, nachvollziehen können.

Haben Sie einmal darüber nachgedacht, was eigentlich die Hauptarbeit eines Netzwerkadministrators ist? Es ist nicht – bzw. sollte nicht – die Installation und Betreuung von Servern sein. Wer regelmäßig einen Dienst oder gar einen Server neu starten muss, weil dieser »sich aufgehängt hat«, sollte schnellstmöglich über einen Ersatz nachdenken. Das Netzwerk sollte so funktionieren, dass Sie eigentlich kaum etwas zu tun haben. Nur dann haben Sie die Zeit und die Ruhe, seine Funktions-

fähigkeit im Auge zu behalten, den regelmäßigen und auch nicht »abschaltbaren« Problemen wahrscheinlich immer der gleichen Anwender freundlich und kompetent zu begegnen und sich auf einen größeren Problemfall vorzubereiten, bei dem Sie, wenn er auftritt, Ihre ganze Kraft und Ihr ganzes Können brauchen und einsetzen müssen, um das Netzwerk am laufen zu halten. Darauf sollten Sie sich vorbereiten und sich Meßinstrumente überlegen, mit deren Hilfe Sie auftretende Probleme schnell eingrenzen können.

7 Einrichtung einer Firewall

Der Begriff »Firewall« gehörte früher nur zum Vokabular von Experten: Er bezeichnet ein Konzept, mit dem die Datennetze von großen Firmen und Behörden vor Industriespionage und Agenten fremder Mächte geschützt wurden. Eine strikte Trennung zwischen internem und externem Netz blockt jeden unerwünschten und potentiell gefährlichen Datenverkehr ab, der von außen Einlass begehrt. Zugleich wird so verhindert, dass aus dem internen Netz unliebsame Verbindungen nach draußen aufgenommen werden, über die womöglich Betriebsgeheimnisse verraten oder private Geschäfte getätigt werden. Ein besonderer, zuverlässiger Router sorgt für die Abschottung der Innen- von der Außenwelt; da er ähnlich wie eine Brandmauer, die zwischen Häusern die Ausbreitung eines Dachstuhlbrands auf das Nachbargebäude verhindern soll, digitale Flächenbrände (sprich: die Ausbreitung von Computerviren und -würmern) begrenzen soll, bezeichnet man ihn auch selbst als Firewall.

Inzwischen treiben nun dermaßen viele unfreundliche Zeitgenossen ihr Unwesen im Internet, dass jeder Rechner, der auch nur gelegentlich mit dem Internet verbunden ist, durch eine solche Firewall vor Angriffen geschützt sein sollte. Die Firewall hat die Aufgabe, alle ein- und ausgehenden Datenpakete zu überprüfen und jedes Paket abzuweisen, das offenkundig unnütz oder unerwünscht und damit potenziell gefährlich ist.

Eine Firewall kann nach wie vor als eigenständiges Gerät zwischen das Internet und das zu schützende interne Netzwerk geschaltet sein, auch wenn letzteres im einfachsten Fall nur einen einzigen Rechner umfasst. Sie kann gleichzeitig als Router arbeiten und dadurch mehrere interne Netze voneinander trennen; diese Funktion findet man bei vielen DSL- und WLAN-Routern, die Datenpakete filtern und damit Firewall-Aufgaben übernehmen können. Schließlich kann sich ein Rechner durch eine »eingebaute Firewall« selbst schützen, wenn das Betriebssystem oder eine speziell installierte Software die Paketfilterung ausführt. Eine solche Anwendung wird in der Windows-Welt oft als persönliche Firewall (»personal firewall«) bezeichnet.

Linux kann in all diesen Einsatzszenarien glänzen: Dank des ins Betriebssystem integrierten Paketfilters kann es sowohl als eigenständige Firewall (mit oder ohne Routingaufgaben) arbeiten als auch die Funktion der persönlichen Firewall übernehmen.

Achtung

Viele Sicherheitsexperten kritisieren die persönlichen, auf dem zu schützenden Rechner implementierten Firewalls, weil sie den Nutzern ein trügerisches Gefühl von Sicherheit vermitteln würden: Anwender neigen dazu, alle Firewall-Beschränkungen zu deaktivieren, sobald sie beim Arbeiten mal als störend empfunden werden, und eventuelle Eindringlinge, die den Rechner hacken, deaktivieren sogleich die Firewall, damit sie nicht bei ihrem bösen Tun hinderlich ist.

Diese Kritik mag auf die typischen Windows-Firewalls zutreffen (obwohl nach unserer Beobachtung auch dort die lokalen Firewalls eher nützen als schaden). Bei einem Linux-System sollten Sie aber die eingebaute Firewall auf jeden Fall aktivieren (zumal sie nichts kostet), denn sie bietet einen zusätzlichen Schutz, indem sie beispielsweise jeglichen fremden Zugriff auf netzwerkfähige Anwendungen verwehrt, die unbeabsichtigt installiert wurden. Außerdem muss ein Hacker erst einmal root-Rechte erlangen, um die Firewall-Einstellungen manipulieren zu können – und das ist auf einem gut gepflegten Linux-System (wie dem Ihren!) schon gar nicht so einfach.

7.1 Was macht ein Paketfilter?

Paketfilterung bedeutet, dass jedes Datenpaket anhand einiger Kriterien überprüft wird, die sich ohne gar zu aufwendige Prüfungen feststellen lassen – schließlich soll die Firewall den Transport der Pakete nicht spürbar verzögern, und die Firewall darf nicht dadurch angreifbar sein, dass sie sich in der logischen Struktur von Anwendungsprotokollen verheddert. (Wir werden aber später sehen, dass man oft von diesen hehren Grundsätzen abweicht und die Firewall doch ein bisschen genauer hinschauen lässt.) Die wichtigsten dieser Kriterien sind:

- die IP-Adresse des Absenders
- die IP-Adresse des Empfängers
- das verwendete Übertragungsprotokoll (meist TCP, UDP oder ICMP)
- die Ziel-Portnummer (für TCP und UDP) bzw. der Nachrichtentyp (ICMP)
- die Flag-Bits (für TCP)
- der Name der Netzwerkschnittstelle, über die das Paket angeliefert wurde bzw. über die es den Rechner verlassen soll

Anhand der Flag-Bits lässt sich bei TCP-Paketen unterscheiden, ob sie die Aufnahme einer neuen Verbindung anzeigen (SYN-Bit gesetzt), ob sie einen solchen Verbindungswunsch bestätigen (SYN- und ACK-Bits gesetzt) oder ob sie zur Datenübermittlung im Rahmen einer bestehenden Verbindung dienen.

Insbesondere die Ziel-Portnummer gibt Auskunft darüber, um was für eine Nachricht es sich handelt, denn sie entscheidet, welches Programm im Zielrechner den

Wunsch zur Verbindungsaufnahme zugestellt bekommt und, falls es ihn akzeptiert, den anschließenden Datenverkehr in dieser Verbindung abhandelt. Es gibt Konventionen darüber, welches Übermittlungsprotokoll über welche zielseitige Portnummer abgewickelt wird; in maschinen- (und menschen-) lesbarer Kurzform finden Sie eine Liste dieser Zuordnungen auf Ihrem Linux-Rechner in der Datei `/etc/services`.

Beispielsweise lauscht ein Webserver im Regelfall auf dem Port 80 auf ankommende Verbindungen und wickelt den Datenverkehr nach den Regeln des Hypertext-Transfer-Protokolls (HTTP) ab, das wiederum auf dem verbindungsorientierten Protokoll TCP/IP basiert. Daher finden Sie in `/etc/services` eine Zeile mit der Portangabe `80/tcp` und den beiden Kurzbezeichnungen `http` und `www`, die für den WWW-Dienst stehen.

Allerdings sind diese Konventionen keineswegs verbindlich: Es ist technisch durchaus möglich, über den TCP-Port 80 E-Mails anstelle von WWW-Seiten zu übertragen oder einen Webserver auf einem abweichenden Port zu betreiben (z. B. sind die Nummern 8000 und 8080 hierfür ganz beliebt, weil die Portnummern unter 1024 auf Unix-Systemen nur von Programmen bedient werden dürfen, die unter der root-Kennung gestartet wurden). Diese Flexibilität hat sicher zum Erfolg der IP-Protokollfamilie beigetragen, jedoch setzt sie der Wirksamkeit von Firewalls Grenzen: Wenn Sie Pakete mit Zielport 80 durch Ihre Firewall hindurchlassen, um den Zugriff aufs Web zu gestatten, kann diese Portfreigabe auch dazu genutzt werden, per Filesharing-Programm die Previews der neuesten Kinofilme durch die Firewall zu schleusen.

Solch ein »Missbrauch« von Ports setzt freilich voraus, dass sich die Client- und die Serverseite über eine abweichende Nutzung geeinigt haben. Deshalb wird in aller Regel die Kommunikation im Internet doch über die gemäß Konvention vorgesehenen Portnummern abgewickelt, und Paketfilter-Firewalls bleiben einstweilen ein probates Mittel, um festzulegen, welche Dienste zwischen welchen Kommunikationspartnern nutzbar sein sollen.

Stateful Filtering

Streng genommen müsste ein Paketfilter über das Schicksal eines jeden Datenpakets allein anhand der genannten Kriterien entscheiden, unabhängig von früher empfangenen Paketen. Eine solche, »stateless« genannte Betriebsweise hat den Vorteil, dass die Datenpakete auch auf wechselnden Wegen durchs Netz geleitet werden können, um beispielsweise gestörte Router und Leitungen zu umgehen oder um zur Erhöhung des Durchsatzes mehrere parallele Verbindungen wechselweise zu nutzen, oder wenn Hin- und Rückweg in einer Verbindung unterschiedliche Wege nehmen (»asymmetrisches Routing«). Das »stateless« Firewalling funktioniert daher auch, wenn ein einzelner Paketfilter nicht alle Pakete zu sehen bekommt, weil ein Teil davon an ihm vorbeiläuft.

Nun werden Sie zu Recht einwenden, dass in Ihrem Netz und mit Ihrer Anbindung ans Internet solch komplizierte Routing-Situationen gewiss nicht auftreten werden. Die stateless-Betriebsweise bringt Ihnen deshalb eher Probleme, denn es ist ziemlich

schwierig, den Paketen anzusehen, welche Rolle sie im Rahmen einer TCP/IP-Verbindung wirklich spielen: Ist ein Paket, das vom Port 80 des externen Webservers www.dubio.us kommt, eine Antwort auf einen Webseitenabruf durch einen Browser, der auf einem PC im lokalen Netz läuft, oder ist es von einem Hacker[1], der den externen Webserver unter seiner Kontrolle hat, zu uns auf den Weg gesandt worden, um Unheil anzurichten?

Viele Paketfilter führen deshalb Buch über die zwischen zwei Partnern ausgetauschten Datenpakete und können dadurch feststellen, ob ein Paket zu einer bereits bestehenden Verbindung gehört und von welcher Seite diese Verbindung initiiert wurde. Dieses Vorgehen wird als »Stateful Filtering« bezeichnet, was man am ehesten als »zustandsbezogene Filterung« eindeutschen könnte und deshalb lieber mit dem englischen Namen benennt. Es funktioniert natürlich nur, wenn der Filter alle relevanten Pakete zu sehen bekommen hat (also nicht, wenn ein Teil von ihnen über andere Wege und ggf. andere Filter transportiert wird) und wenn der Speicherplatz für die interne Buchführung ausreicht, um die Informationen über alle gleichzeitig bestehenden Verbindungen vorzuhalten.

Netfilter: Paketfilter im Linux-Kernel

Linux bietet eine in das Betriebssystem integrierte Paketfilter-Firewall mit Stateful Filtering. Hierzu gehören zum einen die Kernelmodule, die die Paketfilterung beim Empfang, beim Aussenden oder beim Weiterleiten von Datenpaketen vornehmen, und zum anderen das Programm `iptables`, mit dem die Filtertabellen des Kernels verwaltet werden; beides wird vom Netfilter-Projekt[2] entwickelt und ist Bestandteil der gängigen Distributionen. Die Kernelmodule können in den Linux-Kernel fest einkompiliert sein, oder sie werden beim ersten Aufruf von `iptables` automatisch geladen.

Damit das Kommando `iptables` verfügbar ist, muss das passende Paket installiert werden:

Firewall-Administration mit iptables	Debian	SUSE
iptables	iptables	iptables

Tabelle 7.1: Paketübersicht

Helferroutinen für das Connection Tracking

Voraussetzung für das Stateful Filtering ist, dass die Firewall über bestehende Verbindungen und deren Zustand Buch führt, also darüber, ob sich die Verbindung in der Aufbauphase befindet oder als etabliert gilt oder von einem der Partner beendet wurde. Dazu muss sie sämtliche durchlaufenden Nachrichten analysieren

[1] Wir benutzen die Bezeichnung »Hacker« hier für bösartige Netzbewohner. Im Insider-Jargon bezeichnet man diese als »Cracker«; Hacker ist dort eine liebevolle Bezeichnung für versierte, erfahrene und ehrenhafte Bit-Fummler.

[2] `http://www.netfilter.org`

und speziell beim verbindungsorientierten TCP-Protokoll nachvollziehen, in welchen Kommunikationsphasen sich die Partner befinden. Nur dadurch lässt sich entscheiden, ob ein Datenpaket offenkundig zum Zustand der Verbindung passt und die Firewall passieren darf. Ist dies nicht der Fall, muss man davon ausgehen, dass ein böswilliger Dritter manipulierte Pakete auf den Weg geschickt hat, die entweder die Firewall oder den Empfänger täuschen sollen, und solche Nachrichten entsorgt die Firewall am besten, bevor sie beim Empfänger Schaden anrichten können. Diese »Verbindungsverfolgung« wird im Linux-Englisch als »Connection Tracking« bezeichnet.

Leider erschweren einige Kommunikationsprotokolle das Connection Tracking dadurch, dass sie den Informationsaustausch über mehrere separate Verbindungen abwickeln. Beispielsweise nimmt ein FTP-Client Kontakt zum TCP-Port 23 des Servers auf und übermittelt, wenn diese Verbindung zustande gekommen ist, zunächst hierüber die Autorisierungsdaten (Kennung und Passwort, leider beides im Klartext, völlig ungeschützt) und sodann die Aufträge an den Server durch Anweisungen wie `DIR`, `GET` oder `PUT`. Sobald aber die eigentlichen Nutzdaten übertragen werden, baut in der ursprünglichen Fassung des FTP-Protokolls *der Server* eine zusätzliche Verbindung zum Client auf, und zwar zu einem dynamisch festgelegten Port, den der Client ihm durch eine `PORT`-Anweisung mitgeteilt hat.

Für eine Firewall, die den Datenverkehr beobachtet, sieht dies aber so aus, als würde irgendein fremder Rechner den Client durch eine Verbindungsaufnahme zu einem völlig unüblichen Port belästigen wollen, weswegen sie dies unterbindet und der Client auf die Daten vom Server warten kann, bis er schwarz wird.

Es gibt zwei Möglichkeiten, dennoch FTP-Verbindungen durch eine Firewall hindurch aufzubauen[3]: Neue FTP-Programme können die Richtung der Datenverbindung umdrehen (passiver Modus), sodass stets der Client die Initiative übernimmt und in der Firewall der dafür zuständige Zielport (TCP-Port 20) freigegeben werden kann. Dies setzt aber voraus, dass sowohl die Client- als auch die Server-Software diesen Modus beherrschen, und das ist (jedenfalls außerhalb der Linux-Welt) noch immer nicht überall der Fall.

Zum anderen kann auch die Firewall bei der Lösung des Problems helfen: Wenn sie das FTP-Protokoll gut kennt, kann sie alle Verbindungen von Clients zum FTP-Port eines anderen Rechners daraufhin beobachten, ob irgendwann eine `PORT`-Anweisung gesendet wird und anschließend eine dazu passende Verbindungsaufnahme von der anderen Seite eintrifft. Dann handelt es sich offenkundig um eine FTP-Datenverbindung, und die Firewall kann diese zulassen, wenn ihr Regelsatz das vorsieht. Damit auch die Linux-Netfilter-Firewall solche problematischen Protokolle unterstützen kann, bietet der Netfilter-Code eine Plugin-Schnittstelle für sogenannte Helper-Kernelmodule. Diese können vom Administrator bei Bedarf geladen werden und informieren das Connection Tracking, wenn bestimmte IP-Pakete inhaltlich zu einer bereits existierenden Verbindung gehören (mit ihr »related«, d. h. verwandt sind),

[3] Aber bitte nur anonymes FTP! FTP mit Passwort-Autorisierung sollte als Anstiftung zum Datenklau unter Strafe gestellt werden!

obwohl das auf den ersten Blick nicht zu erkennen ist. Das für die Beobachtung der FTP-spezifischen Absonderlichkeiten zuständige Modul heißt `ip_conntrack_ftp`, und wenn es durch das Kommando

```
modprobe ip_conntrack_ftp
```

oder durch Eintragen in eine Konfigurationsdatei wie `/etc/modules` geladen wurde, erfasst das Stateful Filtering auch die FTP-Datenverbindungen.

Auch für einige andere, nicht ganz so populäre Protokolle gibt es spezielle Helper-Module, weil sich deren Eigenheiten durch normale Firewall-Regeln nur schwer erfassen lassen:

- `ip_conntrack_amanda` für die Backup-Software Amanda
- `ip_conntrack_pptp` für PPTP, ein in Windows-Umgebungen häufig eingesetztes Programm zum Tunneln von Verbindungen durch unsichere Netze (zu den unter Linux zu bevorzugenden Lösungen finden Sie mehr in Abschnitt 11.2 auf S. 273).
- `ip_conntrack_tftp` für TFTP (ein stark vereinfachtes File-Transfer-Protokoll, das hauptsächlich zum Booten übers Netzwerk benutzt wird, z. B. von plattenlosen Workstations)
- `ip_netbios_ns` zur Weiterleitung von Broadcast-Nachrichten des Netbios-Nameservice und zum Empfang der zugehörigen Antworten
- `ip_netbios_irc` für IRC (Internet Relay Chat)

7.2 Ein Kommando für (fast) alles: iptables

Nahezu alle firewallrelevanten Einstellungen werden mit Hilfe des `iptables`-Kommandos getroffen. Es steht seit der Linux-Kernelversion 2.4 zur Verfügung und folgt auf frühere Anläufe mit den Kommandos `ipfwadm` (Kernel 2.0) und `ipchains` (Kernel 2.2)[4].

Das `iptables`-Kommando bietet eine Vielzahl von Optionen, und sein Funktionsumfang kann zudem noch durch Erweiterungsmodule ausgebaut werden, mit denen zusätzliche Kriterien für die Paketauswahl verfügbar werden. Zu jeder solchen Erweiterung gehört jeweils ein Kernelmodul, das diese Kriterien auf die zu filternden Datenpakete anwendet. Ein flüchtiger Blick auf die Manpage von `iptables` zeigt schon, dass dieses Kommando zwar sehr mächtig, aber auch etwas unübersichtlich ist. Es weist darüber hinaus einige Besonderheiten auf, die dem Firewall-Neuling das Leben schwer machen können:

- Die Reihenfolge der Optionen ist nicht beliebig; beispielsweise kann eine Zielportnummer mit `--dport 80` als Kriterium nur spezifiziert werden, wenn *zuvor* durch `-p tcp` oder `-p udp` eines der Protokolle TCP oder UDP ausgewählt wurde. (Dies liegt daran, dass erst durch die Angabe des Protokolls das Erweiterungsmodul aktiviert wird, das anschließend die Portnummer »verstehen« kann.)

[4] Es herrschte allgemeine Verwunderung, als der Kernel 2.6 kein neues Firewall-Kommando mit sich brachte.

- Die mit `--` beginnenden Optionsnamen dürfen nicht (wie bei anderen Kommandos oft üblich) abgekürzt werden; ähnlich lautende Namen (wie `--dport` und `--dports`) haben oft ganz unterschiedliche Bedeutungen. Auch darf kein Gleichheitszeichen zwischen dem Namen und dem Wert einer Option stehen: `--dport 80` ist korrekt, `--dport=80` wird nicht akzeptiert.
- Fehlermeldungen des Kommandos bei Syntax- oder logischen Fehlern sind oft irreführend:

```
hugo:~# iptables -A FORWARD -p tcp --dport 80 -j ACCEPT
hugo:~# iptables -A FORWARD -p tcp -dport 23 -j ACCEPT
Bad argument '23'
Try 'iptables -h' or 'iptables --help' for more information.
hugo:~#
```

  Das erste Kommando ist korrekt, beim zweiten fehlt ein Minuszeichen vor dem Parameternamen `dport`. Deshalb kommt der Argumentparser aus dem Tritt und kann mit der `23` nichts anfangen – der Nutzer, dem der Vertipper widerfahren ist, wird allerdings lange darüber sinnieren, was an diesem Wert falsch sein soll.
- `iptables` analysiert die Benutzereingaben, baut eine Datenstruktur auf und versucht anschließend, diese an den Netfilter-Code im Kernel zu übergeben, der im Kerneladressraum eine äquivalente Struktur anlegen soll. Dies kann zum einen recht lange dauern, wenn bereits eine große Anzahl von Regeln aktiv ist (weil der Regelsatz im Speicher komplett neu aufgebaut wird). Zum andern »weiß« das Kommando nicht, welche Erweiterungsmodule für den Kernel vorhanden sind. So kann es passieren, dass eine Option auf der Kommandozeile akzeptiert wird, vom Kernelcode aber zurückgewiesen wird (oder nicht per Kommando eingegeben werden kann, obwohl das Kernelmodul bereit steht). Bisweilen werden sogar von den Distributionen solche Diskrepanzen zwischen Kernel- und Userspace eingerichtet – sicher unfreiwillig, aber für die Anwender auf jeden Fall störend.

7.2.1 Regeln an die Kette gelegt

Die Regeln, nach denen über Annahme oder Zurückweisung eines Datenpakets entschieden wird, sind sequenziell als »Regelketten« (engl. »chains«) organisiert, die von jedem Datenpaket nach dem Vorbild des Spießrutenlaufens passiert werden müssen: Es wird der Reihe nach für jede Regel geprüft, ob ihre Bedingungen auf das Datenpaket zutreffen. Wenn ja, kommt es zur Ausführung einer Aktion (im Englischen als »target«, Ziel, bezeichnet), die dieser Regel zugeordnet ist. Die wichtigsten Aktionen sind:

- ACCEPT – das Paket akzeptieren
- DROP – das Paket verwerfen.

Mit der Ausführung einer solchen Aktion ist das Spießrutenlaufen für dieses Paket beendet; die weiteren Filterregeln können an der Entscheidung nichts mehr ändern und werden deshalb gar nicht mehr geprüft. Ansonsten werden die nächsten Regeln aus der Kette der Reihe nach herangezogen, um über das Schicksal des Pakets

zu befinden, bis das Ende der Kette erreicht ist. Das letzte Wort hat dann schließlich die der Kette zugeordnete Policy: Sie bestimmt, welche Aktion für Pakete auszuführen ist, die von keiner Regel erfasst wurden, und ist standardmäßig auf ACCEPT eingestellt.

7.2.2 Gegenverkehr: INPUT- und OUTPUT-Regeln

Eigentlich würde eine globale Regelkette ausreichen, um sämtliche Paketfilterregeln aufzunehmen, da alle wesentlichen Attribute eines Pakets durch Bedingungen in jeder Regel abfragbar sind. In der Praxis sind aber die Regeln für ankommende Pakete oft sehr viel restriktiver ausgelegt als die Regeln für abgehende Pakete, weil man ja zum eigenen Rechner viel mehr Vertrauen hat als zum Rest der Welt. Nahezu jede Filterregel würde eine Bedingung enthalten, die abfragt, ob das Paket draußen oder drinnen generiert wurde. Um diese Umständlichkeit zu vermeiden und damit auch eine absehbare Fehlerquelle bei der Regelerstellung auszuschalten, leitet der Linux-Netfilter hereinkommende und hinausgehende Pakete durch völlig getrennte Regelketten, die mit den nicht gerade überraschenden Namen INPUT bzw. OUTPUT benannt sind.

Dadurch erhält auch jede Richtung ihre eigene Policy. Setzt man die Policy der INPUT-Kette auf DROP, so hat man schon eine vernünftige Grundsatzentscheidung getroffen – nämlich alle von außen kommenden Datenpakete abzulehnen – und muss sich nur noch um die vergleichsweise wenigen Ausnahmen kümmern.

Für den Fall, dass der Computer Daten an sich selbst schickt (z. B. über die Adresse 127.0.0.1 an das Loopback-Interface), wird so verfahren, als würden die Daten vom Netzwerk gespiegelt: Die Pakete durchlaufen die OUTPUT- und dann noch zusätzlich die INPUT-Regeln.

Üblicherweise werden die Regelketten sequenziell aufgebaut. Durch das Kommando `iptables -A INPUT ...` wird eine Regel an die INPUT-Kette angefügt (das `A` steht für »append«). Mit der Option -L (wie »list«) können Sie eine Kette oder, wenn der Kettenname fehlt, alle Regelketten auflisten lassen:

```
hugo:~# iptables -A INPUT -p tcp --dport 22 -j ACCEPT
hugo:~# iptables -L INPUT -nv
Chain INPUT (policy ACCEPT 4 packets, 392 bytes)
```

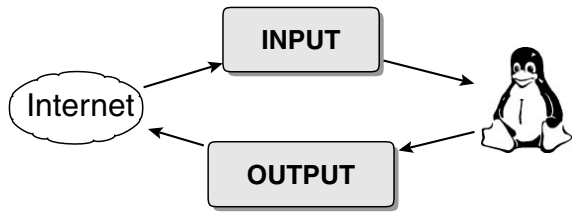


Abbildung 7.1: Paketfluss durch INPUT und OUTPUT

```
 pkts bytes target     prot opt in   out   source      destination
    0     0 ACCEPT     tcp  --  *    *     0.0.0.0/0   0.0.0.0/0     tcp dpt:22
hugo:~# iptables -L -nv
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
hugo:~#
```

Unseligerweise will `iptables` in der Ausgabe sämtliche IP-Adressen durch passende Namen ersetzen. Da es hierzu das Domain Name System befragt, aber für viele der in den Regeln verwendeten Adressen üblicherweise keine Antwort erhält, nimmt dies beträchtliche Zeit in Anspruch. Wenn Sie also nicht ohnehin nebenbei Kaffee trinken und Kuchen essen und sich daher über jede Verzögerung freuen, sollten Sie den `iptables -L`-Kommandos immer die Option `-n` (numerische Ausgabe von Adressen und Portnummern) hinzufügen. Tippen Sie dann noch ein `v` dazu, erhalten Sie zusätzlich die Anzahl von Paketen und deren akkumulierte Längen, auf die die Regel zugetroffen hat[5].

In der iptables-Ausgabe ist zu erkennen, dass für die Policy für die INPUT-Kette noch die Voreinstellung ACCEPT gilt. Die im obigen Beispiel erzeugte Firewall-Regel, die alle ankommenden Secure-Shell-Verbindungen zulässt (denn diese benutzen normalerweise den »destination port«, also den Zielport 22), ist deshalb wirkungslos. Erst wenn durch die Policy Pakete abgelehnt werden, können Sie mit ACCEPT-Regeln diejenigen Protokolle auswählen, die Sie zulassen wollen:

```
hugo:~# iptables -P INPUT DROP
hugo:~# iptables -L INPUT -nv
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in   out   source      destination
    0     0 ACCEPT     tcp  --  *    *     0.0.0.0/0   0.0.0.0/0     tcp dpt:22
hugo:~#
```

Jetzt kommen vom Netz nur noch ssh-Pakete herein. Für praktische Anwendungen ist das ein bisschen wenig, denn Sie wollen auch noch anderes ins System lassen, beispielsweise die Antworten auf Webseitenaufrufe Ihres Browsers – wir werden daher den Regelsatz später noch erweitern. Aber durch das Setzen einer DROP-Policy für die INPUT-Kette haben Sie schon den wichtigsten Schritt getan, um Ihr System zu schützen.

[5] Die großbuchstabigen Aktionsoptionen des iptables-Kommandos lassen sich mit den sonstigen (kleinbuchstabigen) Optionen nicht hinter einem gemeinsamen Minuszeichen zusammenfassen!

Ein wichtiger Hinweis scheint uns an dieser Stelle angebracht: Sie können mit Firewall-Regeln durchaus herumexperimentieren und die Auswirkungen verschiedenster `iptables`-Kommandos studieren. Am interessantesten ist es natürlich, wenn Sie im lokalen Netz einen zweiten Rechner haben, von dem aus Sie testen können, welche Services (Secure Shell, Webserver, Ping, ggf. E-Mail) noch erreichbar sind und welche durch die Firewall-Regeln blockiert werden. Solange Sie die Regeln nur dynamisch durch `iptables` erzeugen, ist es auch nicht allzu schlimm, wenn Sie den Regelsatz »verkorksen«, denn erstens können Sie einzelne und auch alle Regeln wieder löschen (dazu gleich mehr), und zweitens gelten die Regeln nur für die aktuelle Session und sind – solange Sie keine besonderen Vorkehrungen treffen – nach einem Reboot wieder weg. Nur wenn das System außerhalb Ihrer Reichweite steht und Ihr Zugang zum Rechner ausschließlich übers Netz erfolgt (und damit die Firewall überwinden muss), sollten Sie sich gut überlegen, was Sie tun …

Streng genommen ist die Zuordnung einer Policy zu einer Regelkette überflüssig: Man könnte ebenso gut eine der Policy entsprechende Regel, die keine sonstige Bedingung enthält, ans Ende der Kette stellen. Jedes Paket, das die davor stehenden Regeln nicht erfasst haben, würde durch diese finale Regel akzeptiert oder verworfen. Doch könnte man dann keine Filterregeln mehr einfach ans Ende der Kette anhängen, weil sie dort wirkungslos blieben. Die Policy-Vereinbarung hingegen wirkt immer erst, nachdem die letzte Regel in der Kette durchlaufen wurde, und es können bei Bedarf weitere Regeln problemlos hinten angefügt werden.

7.2.3 Durchgangsverkehr: FORWARD-Regeln

Zu den INPUT- und OUTPUT-Regelketten kommt noch eine weitere hinzu, die von all denjenigen Datenpaketen durchlaufen wird, die weder in den betreffenden Rechner hineingehen noch von ihm erzeugt wurden – solche kann es nur geben, wenn der Rechner im Netz Routing-Aufgaben übernimmt, also wie ein Router verschiedene Netze miteinander verbindet. Damit sich Linux-Systeme nicht durch Übereifer im Netzwerk unbeliebt machen, spielen sie sich nach dem Booten nicht freiwillig als Router auf. Erst wenn man sie dazu auffordert, indem man eine Eins in die Pseudodatei `/proc/sys/net/ipv4/ip_forward` schreibt, fühlen sie sich für Routing-Aufgaben zuständig:

```
echo 1 >/proc/sys/net/ipv4/ip_forward
```

Alternativ kann man dies auch durch einen Aufruf des Programms `sysctl`

```
sysctl -w net.ipv4.ip_forward=1
```

einschalten oder, falls die Distribution eine Datei `/etc/sysctl.conf` vorsieht, eine Zeile mit dem Inhalt

```
net.ipv4.ip_forward=1
```

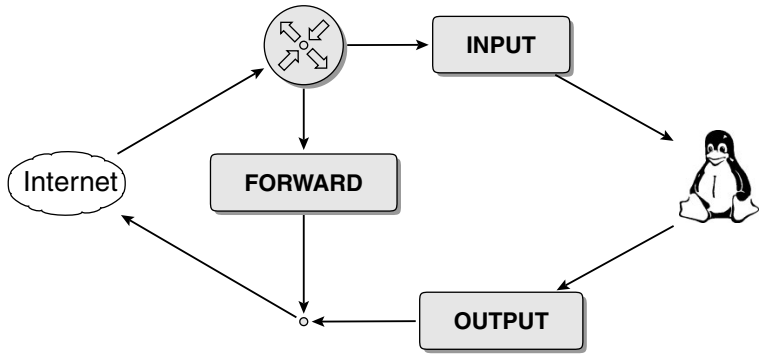


Abbildung 7.2: Paketfluss durch INPUT, OUTPUT und FORWARD

dort eintragen[6]. Wenn der betreffende Rechner Datenpakete empfängt, die laut Empfängeradresse nicht für ihn bestimmt sind (Näheres hierzu in Abschnitt A.1.2 auf S. 326), würde er diese normalerweise ignorieren. Wenn aber das Forwarding auf eine dieser Arten eingeschaltet ist, versucht er, die Pakete – in der Regel über eine andere Netzwerkkarte – wieder loszuwerden, auf dass sie dort den richtigen Empfänger finden mögen, und genau das ist die Hauptbeschäftigung eines Routers.

Auch dieses »Forwarding« kann durch Netfilter-Regeln gesteuert werden, und es gibt hierfür eine eigene Regelkette mit dem wiederum naheliegenden Namen FORWARD.

Die Abbildung soll deutlich machen, dass die über eine Netzwerkschnittstelle ankommenden Pakete zuerst einer Routing-Entscheidung unterworfen werden. Stellt sich dabei heraus, dass sie für dieses Linux-System bestimmt sind, werden sie durch die INPUT-Filterkette geschickt und sodann, falls sie diese überlebt haben, einem lokalen Prozess zugestellt. Andernfalls durchlaufen sie die FORWARD-Kette und verlassen im Erfolgsfall das System über eine andere Schnittstelle zum Internet (oder einem sonstigen Netzwerk), genau wie die durch die OUTPUT-Kette geläuterten Pakete, die vom lokalen System in die Welt geschickt werden.

Bei eingeschaltetem Routing und versehen mit geeigneten FORWARD-Regeln kann ein Linux-Rechner somit die Funktion einer hauptberuflichen Firewall übernehmen, nämlich verschiedene Netze miteinander verbinden und kontrollieren, welche Daten zwischen diesen Netzen ausgetauscht werden dürfen.

7.3 iptables-Kommandosyntax

Das Kommando `iptables` verfügt über eine große Anzahl möglicher Optionen; die Manpage wirkt daher auf den ersten Blick erschreckend umfangreich. Schaut man

[6] Es gibt außerdem für jede Netzwerkschnittstelle eine eigene Pseudodatei, z. B. `/proc/sys/net/ipv4/conf/eth0/forwarding` für `eth0`, durch die das Routing über diese Schnittstelle gezielt ein- oder ausgeschaltet werden kann.

aber genauer hin, so lassen sich diese Optionen nach ihrem Zweck in sechs Gruppen einteilen, und diese müssen nicht immer vertreten sein:

- Auswahl der Regeltabelle
- Auswahl der Regelkette innerhalb der Tabelle und der Operation, die auf der Kette auszuführen ist
- Selektion von Paketen nach Adresse und Interface
- Protokollspezifikation
- Match-Erweiterungen (Module und dazu gehörende Parameter für spezielle Filterungen)
- Aktion, die auf die von der Regel erfassten Datenpakete anzuwenden ist, und zur Aktion gehörende Parameter

Wir empfehlen Ihnen, die Optionen in der Reihenfolge dieser Gruppen anzugeben, die wir nachfolgend näher beschreiben. Das ist zwar nicht immer zwingend erforderlich, macht das Kommando aber übersichtlicher und erspart es Ihnen, darüber nachzudenken, ob eine andere Reihenfolge überhaupt zulässig wäre. Bei der Darstellung von Anwendungsfällen in den folgenden Abschnitten geben wir noch detailliertere Informationen zu einzelnen Optionen, doch erschöpfend behandeln können wir sie hier nicht, denn Sie haben schließlich kein Buch über das `iptables`-Kommando gekauft. Bei Bedarf kann Ihnen die einschlägige Literatur sicher weiterhelfen:

Weitere Literatur

- `man iptables`
- Netfilter-HOWTO
- Ralf Spenneberg: »Linux-Firewalls mit iptables & Co.« Addison-Wesley, 2006, ISBN 3-8273-2136-0

Auswahl der Regeltabelle

Der Linux-Kernel verwaltet die Netfilter-Regeln in vier separaten Tabellen. Die für das Firewalling wichtigste Tabelle trägt den Namen `filter` und nimmt, wie der treffend gewählte Name andeutet, die Regeln zum Filtern der IP-Pakete auf. Die Tabelle `nat` dient dazu, mittels besonderer Regeln die Quell- und/oder Zieladressen von Paketen umzuschreiben (»network address translation«; mehr dazu in Abschnitt 7.5.1 auf S. 156).

Darüber hinaus gibt es die Tabellen `mangle` zur Manipulation von IP-Paketen (beispielsweise zum Setzen von Markierungen oder des Quality-of-Service-Indikators) und `raw` (zum Ausschluss bestimmter Pakete von der Verfolgung des Verbindungsstatus), auf die wir aber nicht weiter eingehen.

Die Syntax dieser Option lautet:

```
{ -t | --table } { filter | nat | mangle | raw }
```

Der Wert `filter` ist voreingestellt, deshalb darf die Option bei der Verwaltung normaler Filterregeln weggelassen werden.

Operationscode und Kettenauswahl

Ein Großbuchstabe markiert die wichtige Rolle dieser Option[7]: Sie gibt an, was das `iptables`-Kommando überhaupt tun soll. Normalerweise werden beim Aufbau des Regelwerks die Regeln der Reihe nach (per Prozedur) an die Regelketten angehängt (`-A`). Die anderen Operationen dienen vorwiegend der manuellen Pflege des Regelsatzes: Im laufenden Betrieb können einzelne Regeln eingefügt (`-I`), ersetzt (`-R`) und gelöscht (`-D`) werden, und damit man nicht den Überblick verliert, kann man jederzeit den aktuellen Stand einzelner oder aller Ketten auflisten lassen (`-L`). Beim Einfügen und beim Ersetzen muss die zu ändernde Position innerhalb der Kette durch eine Nummer identifiziert werden (zur Orientierung können die stets fortlaufend vergebenen Regelnummern beim Auflisten angezeigt werden).

Das Löschen einer Regel kann wahlweise durch Angabe der Nummer oder durch Wiederholung der Regeldefinition, wie sie beim Eintragen formuliert wurde, verlangt werden. Letzteres ist vor allem dann hilfreich, wenn man feststellt, dass eine soeben erzeugte Regel nicht richtig funktioniert: Aus dem Kommandogedächtnis der Shell holt man das `iptables`-Kommando zurück, mit dem die Regel eingetragen wurde, und ersetzt die Operation (meist das `-A`) durch `-D`. Nun genügt es, die Eingabetaste zu drücken, und schon ist die Regel wieder weg.

Die anderen Operationen (`L`, `Z`, `P`, `F`, `N`, `E` und `X`) wirken jeweils auf eine oder (wenn der Name weggelassen wird) auf mehrere gesamte Regelketten und vertragen sich daher nicht mit den weiteren Optionen, mit denen einzelne Regeln aufgebaut werden:

- `{-A | --append} <name>`
  Eine neue Firewall-Regel erzeugen und an die angegebene Kette anhängen.
- `{-D | --delete} <name> [<position>]`
  Eine Regel löschen, die entweder durch ihre Positionsnummer oder durch Wiederholung der Parameter spezifiziert wird, die bei ihrer Erzeugung angegeben wurden.
- `{-I | --insert} <name> [<position>]`
  Die in den weiteren Parametern dieses Kommandos spezifizierte Regel in die Kette `<name>` an der angegebenen Position einfügen (wenn die Positionsangabe fehlt, am Beginn der Kette).
- `{-R | --replace} <name> <position>`
  Die Regel an der angegebenen Position durch eine neue ersetzen, die sich aus den folgenden Parametern ergibt.
- `{-L | --list} [<name>]`
  Regeln auflisten; wenn `<name>` nicht angegeben ist, alle Regeln der ausgewählten Tabelle. Folgende Optionen sind nützlich im Zusammenhang mit `-L`:

[7] Alternativ sind auch »sprechende« Namen erlaubt.

  - `-n` oder `--numeric`: Adressen und Portnummern numerisch anzeigen (es ist immer ratsam, diese Option anzugeben)
  - `-v` oder `--verbose`: Detailliertere Ausgabe, enthält insbesondere die Anzahl der Pakete und die Summe ihrer Längen, die von jeder Regel erfasst wurden.
  - `-x` oder `--exact`: Die genannten Zähler werden auch bei großen Zahlenwerten exakt ausgegeben (andernfalls werden die üblichen Suffixe `K`, `M` und `G` benutzt)
  - `--line-numbers`: Die Regeln jeder Kette werden in der Ausgabe fortlaufend durchnummeriert (damit anschließend gezielt Regeln gelöscht oder an einer bestimmten Stelle eingefügt werden können).

- `{-Z | --zero} [<name>]`
  Paket- und Bytezähler auf null setzen; diese Operation lässt sich sinnvoll mit der Abfrage dieser Zähler verbinden:

  ```
  iptables -L OUTPUT -Z -nv
  ```

  Dieses Kommando listet die Regeln der OUTPUT-Kette mitsamt den aktuellen Zählerständen auf und setzt die Zähler gleichzeitig auf null zurück. Dadurch wird vermieden, dass zwischen der Abfrage und dem Löschen der Zähler weitere Pakete ungezählt durchschlüpfen.

- `{-P | --policy} <name> <target>`
  Policy für eine Kette festlegen (hat dieselbe Wirkung, als sei immer eine Regel mit der Aktion `-j <target>` am Schluss der Kette eingetragen).

- `{-F | --flush} [<name>]`
  Alle Regeln dieser Kette löschen (wenn kein `<name>` angegeben ist, alle Regeln der angesprochenen Tabelle löschen). Die Policy bleibt hierdurch unbeeinflusst.

- `{-N | --new-chain} <name>`
  Eine neue, benutzerdefinierte Regelkette anlegen.

- `{-E | --rename-chain} <alternname> <neuername>`
  Eine benutzerdefinierte Regelkette umbenennen (Aufrufe dieser Kette werden automatisch angepasst).

- `{-X | --delete-chain} <name>`
  Gesamte Regelkette löschen (nur möglich für selbst definierte Ketten, die nicht referenziert werden).

Allgemeine Selektoren

Unabhängig vom verwendeten Protokoll sind einige Selektionskriterien auf alle Arten von IP-Paketen anwendbar. Hierzu gehören die Quell- und die Zieladresse:

- `{-s | --source} [!] <adresse>[/<netzmaske>]`
  Absenderadresse bzw. -netz

- `{-d | --destination} [!] <adresse>[/<netzmaske>]`
  Zieladresse bzw. -netz

Eine Netzangabe kann durch eine klassische Netzmaske oder durch die Präfixlänge (CIDR[8]-Notation) dargestellt werden: `192.168.99.0/255.255.255.0` und `192.168.99.0/24` haben dieselbe Bedeutung und erfassen alle Adressen, die in den linken drei Bytes mit der Netzwerkadresse übereinstimmen, also den Adressbereich von `192.168.99.0` bis `192.168.99.255`.

Steht ein Ausrufezeichen zwischen dem Optionsnamen und dem Wert, so werden dadurch die Pakete selektiert, die *nicht* von der Adressangabe erfasst werden: Durch

```
-s ! 192.168.99.0/24
```

werden diejenigen Pakete ausgewählt, die nicht aus dem 24-Bit-Subnetz `192.168.99.0` kommen.

Ebenfalls protokollunabhängig ist die Abfrage nach dem Interface, über das die Pakete angekommen sind oder den Rechner verlassen werden:

- `{-i | --in-interface} [!]`
  Eingangsinterface
- `{-o | --out-interface} [!]`
  Ausgangsinterface

Zu beachten ist dabei jedoch, dass die Interface-Information nicht immer verfügbar oder relevant ist: In der OUTPUT-Kette darf das Eingangsinterface nicht abgefragt werden (weil die Pakete nicht über ein solches hereingekommen sind), und analog gibt es in der INPUT-Kette kein Ausgangsinterface.

Die Bedeutung der Selektion kann auch hier durch ein `!` invertiert werden.

Protokollspezifikation

Wichtige Kriterien, die die Nutzanwendung einer IP-Verbindung charakterisieren und dementsprechend häufig bei der Filterung abgefragt werden, sind das benutzte Übertragungsprotokoll und seine Parameter. Bei den Protokollen TCP und UDP sind es insbesondere die Portnummern auf der Serverseite, die für bestimmte Anwendungen typisch sind. Es gibt Standards, Konventionen und darüber hinaus individuelle Festlegungen, über welche Portnummer ein Server seine Dienste anbietet; die gängigen Zuordnungen finden Sie beispielsweise in der Datei `/etc/services`. Schauen wir dort einmal nach Einträgen für HTTP, das Anwendungsprotokoll[9]:

```
victor@hugo:~$ grep -i http /etc/services
# Updated from http://www.iana.org/assignments/port-numbers and other
# sources like http://www.freebsd.org/cgi/cvsweb.cgi/src/etc/services .
```

[8] Classless Inter-Domain Routing

[9] Leider wird der Begriff »Protokoll« hier in zwei unterschiedlichen Bedeutungen verwendet, nämlich zum einen für die auf IP aufsetzende Datenübertragung z. B. über TCP oder UDP (»Layer-4-Protokoll« im OSI-Schichtenmodell) und zum anderen für die Regeln, nach denen Programme miteinander kommunizieren, z. B. über FTP oder HTTP (»Layer-7-Protokoll«).

```
www             80/tcp          http            # WorldWideWeb HTTP
https           443/tcp                         # http protocol over TLS/SSL
https           443/udp
victor@hugo:~$
```

Die ersten beiden Trefferzeilen gehören zu einem Kommentar, der wie üblich durch \#-Zeichen in jeder Zeile gekennzeichnet wird. An den folgenden Trefferzeilen erkennen Sie den Aufbau der Einträge in der `services`-Datei: In der ersten Spalte steht eine Abkürzung für das Anwendungsprotokoll, also den Dienst, und in der zweiten Spalte stehen die Portnummer und `tcp` oder `udp`. Fakultativ können danach weitere Namen für denselben Dienst folgen sowie, wiederum mit \# eingeleitet, ein erklärender Kommentar.

Wir sehen hier, dass HTTP üblicherweise über TCP und Port 80 abgewickelt wird (wegen der Popularität der Abkürzung WWW für das World Wide Web steht hier diese in Spalte 1, und der korrekte Name folgt erst in der dritten Spalte). Das `grep`-Kommando hat außerdem zwei Eintragungen für HTTPS, die kryptografisch gesicherte Variante von HTTP, gefunden, die man auch in den Firewall-Regeln berücksichtigen sollte, wenn der Zugriff zu einem Webserver erlaubt wird – wegen der höheren Sicherheit, die HTTPS bietet, sollten wir uns über dessen Nutzung freuen und dieses Protokoll nicht abblocken!

Einige Anwendungen kommunizieren je nach Lust und Laune (oder vielleicht nach Bedarf) per TCP oder UDP, und es finden sich dann zwei entsprechende Zeilen in `/etc/services`. Für HTTPS ist die Nutzung von UDP allerdings völlig unüblich; wirklich relevant sind beide Protokolle aber für DNS (das Domain Name System, wie Sie spätestens seit S. 23 wissen). Suchen Sie in `/etc/services` nach DNS, so werden Sie eine Enttäuschung erleben; die Herrscher über diese Datei haben den Namensdienst nicht unter seinem Kürzel, sondern unter »domain« eingetragen:

```
victor@hugo:~$ grep -i dns /etc/services
victor@hugo:~$ grep -i domain /etc/services
domain          53/tcp          nameserver      # name-domain server
domain          53/udp          nameserver
victor@hugo:~$
```

Mit dem Wissen um Protokolle und (Ziel-)Portnummern können Sie nun passende Firewall-Regeln formulieren, beispielsweise für die OUTPUT-Kette, um Zugriffe aus dem Linux-System heraus auf beliebige Web- und DNS-Server zu erlauben:

```
iptables -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT
iptables -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 53 -j ACCEPT
```

Die Optionen `--dport` (oder, deutlicher und umständlicher, `--destination-port`) für den Zielport und `--sport` (oder `--source-port`) für den Quellport der Verbindungen sind

nur sinnvoll für die Protokolle TCP und UDP und dürfen deshalb nur *nach* einer Option `-p tcp` oder `-p udp` angegeben werden!

Vor der Portnummer darf wieder ein Ausrufezeichen stehen, sodass die Abfrage für alle Portnummern ungleich der angegebenen erfüllt ist, und auch die Angabe von Portbereichen ist möglich: `--dport 6000:6010` erfasst alle Zielports von 6000 bis einschließlich 6010.

Es ist übrigens auch zulässig, statt der Portnummern Namen von Diensten anzugeben, die vom `iptables`-Kommando dann anhand der `services`-Datei in die passenden Nummern übersetzt werden. Wir raten allerdings davon ab, weil zum einen die Einträge in dieser Datei von Distribution zu Distribution unterschiedlich sein und zum andern sich auch mal durch ein Upgrade ändern könnten, und dann funktionieren plötzlich Ihre Firewall-Regeln nicht mehr.

Eigentlich werden auch die Protokolle durch Nummern bezeichnet (und in den Datenpaketen codiert), doch sind zumindest für die gängigen Protokolle die Namen aus Gründen der Lesbarkeit zu bevorzugen. Die Zuordnung der Namen zu Nummern ist ebenfalls durch eine Datei (`/etc/protocols`) gegeben. Deren Einträge bezeichnen größtenteils exotische Protokolle, denen Sie in freier Wildbahn kaum jemals begegnen werden (außer vielleicht, wenn Sie Firewall-Regeln für virtuelle private Netze schreiben). Wichtig für eine Normalverbraucher-Firewall ist neben TCP und UDP lediglich noch das ICMP-Protokoll (»Internet Control Message Protocol«), das primär dazu gedacht ist, Fehlerereignisse zu übermitteln. Wird beispielsweise versucht, eine Verbindung zum TCP-Port 80 eines Rechners aufzubauen, auf dem gar kein Webserver läuft, so wird dieser Rechner mit einer ICMP-Nachricht antworten, die keine richtigen Nutzdaten enthält, sondern in zwei Bytes das Ereignis »port unreachable« verschlüsselt. Der Möchtegern-HTTP-Client weiß daraufhin, dass er nicht länger auf das Zustandekommen einer TCP-Verbindung warten muss, und kann sogar dem Anwender den Grund in geeigneter Form präsentieren.

Durch das Connection Tracking des Netfilters werden solche Ereignismeldungen als einer Verbindung zugeordnet (»related«) erkannt. Es gibt allerdings noch eine andere nützliche Anwendung von ICMP: Mit dem Kommando `ping` können ICMP-Pakete mit dem Typcode `echo-request` an einen Rechner gesandt werden, der darauf mit jeweils einem ICMP-Paket des Typs `echo-reply` antwortet (man spricht aus naheliegendem Grund auch von Ping- und Pong-Paketen). Damit wird die Funktionsfähigkeit der IP-Verbindung überprüft. Wollen Sie beispielsweise das Ping aus dem lokalen Netz `192.168.42.0/24` zulassen, können Sie die Option `--icmp-type` nutzen, um den passenden Pakettyp zu selektieren:

```
iptables -A INPUT -s 192.168.42.0/24 -p icmp --icmp-type echo-request -j ACCEPT
```

Welche ICMP-Typen `iptables` sonst noch kennt, lässt sich mit

```
iptables -p icmp --help
```

abfragen.

Match-Erweiterungen

Match-Erweiterungen werden durch spezielle Kernelmodule realisiert, die – wenn sie denn überhaupt installiert sind – durch Angabe einer Option `-m` oder `--match` im `iptables`-Kommando automatisch geladen werden und zusätzliche Filterbedingungen bereitstellen. In der Regel gehören zu jeder solchen Erweiterung besondere Optionen, die wiederum nur *nach* der Auswahl der Erweiterung im selben Kommando aufzuführen sind. Beispielsweise lädt das Kommando

```
iptables -A FORWARD -m addrtype --dst-type BROADCAST -j DROP
```

das Kernelmodul `ipt_addrtype.ko`, das den Adresstyp eines jeden Pakets bestimmt und in diesem Fall einen Treffer signalisiert, wenn es sich um ein Broadcast-Paket handelt (wodurch in diesem Beispiel die Weiterleitung von Broadcasts blockiert wird). Eine besonders nützliche Erweiterung wird durch `-m state` aktiviert: Sie überprüft den vom Connection Tracking bestimmten Zustand der Verbindung, zu der das Paket gehört, und erlaubt die Auslösung der zur Filterregel gehörenden Aktion in Abhängigkeit von der Auswahl eines oder mehrerer Zustände durch die Option `--state`:

- `NEW`: Das Paket eröffnet eine neue Verbindung, es gibt keinen Eintrag in der Connection-Tracking-Tabelle, der dazu passt.
- `ESTABLISHED`: Das Paket gehört zu einer Verbindung, innerhalb derer schon Pakete (in beiden Richtungen) beobachtet wurden.
- `RELATED`: Das Paket gehört zwar nicht zu einer bestehenden Verbindung, steht aber mit einer solchen in engem Zusammenhang. Hierzu gehören beispielsweise das Eröffnen einer FTP-Datenverbindung zu einer schon etablierten Kontrollverbindung oder eine ICMP-Nachricht, die offenbar durch zuvor beobachtete Pakete ausgelöst wurde.
- `INVALID`: Das Paket konnte keiner Verbindung zugeordnet werden, hätte aber nach den IP-Konventionen auch nicht aus heiterem Himmel erscheinen dürfen. Entweder verhält sich jemand nicht regelkonform (und führt vielleicht Böses im Schilde), oder das Connection Tracking hat die dazugehörenden Pakete nicht zu sehen bekommen.

Die wichtigste Nutzanwendung des `state`-Matches besteht darin, in einer Regelkette alle diejenigen Pakete zu akzeptieren, die zu bestehenden Verbindungen gehören, und anschließend nur noch Filterregeln zu definieren, die sich um die neu aufgenommenen Verbindungen kümmern. Auch im Interesse einer Verkürzung der Pfadlängen bei der Paketfilterung sollte also am Anfang der INPUT-Kette eine solche Regel erzeugt werden:

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Dadurch werden nach einer positiven Entscheidung über eine Verbindungsaufnahme alle zu dieser Verbindung gehörenden hereinkommenden Pakete sofort akzeptiert. Genauso wird man meist auch für die OUTPUT- und FORWARD-Ketten verfahren.

Aktion

Den Höhepunkt eines `iptables`-Kommandos stellt zweifelsohne der Aktionsteil dar. Er entscheidet im Normalfall darüber, was mit dem Paket geschehen soll, das von der Firewall-Regel erfasst wurde. Die Aktionen (Targets) DROP und ACCEPT haben wir schon vorgestellt, einige weitere folgen im nächsten Abschnitt.

Wie die Match-Erweiterungen können auch Aktionen durch spezielle Kernelmodule realisiert werden. Einige werden mit dem Standardkernel ausgeliefert, und Quellen für weitere (von unterschiedlicher Nützlichkeit und Qualität) sind über die Netfilter-Projektseiten im Netz zu finden.

Ebenfalls in Analogie zu den Match-Erweiterungen können Aktionen zusätzliche Optionen aufweisen, die wiederum *nach* der Aktion im `iptables`-Kommando spezifiziert werden müssen.

Eine einzelne Firewall-Regel darf in keinem Fall mehr als eine Aktion enthalten. Wenn Sie mehrere Aktionen kombinieren wollen, müssen Sie mehrere Regeln (mit gleichem Bedingungsteil) definieren.

Tipp

Wie im absurden Theater kann auch hier der Höhepunkt fehlen: Eine Firewall-Regel darf durchaus ohne Aktion geschrieben werden. Trifft ein Datenpaket auf so eine Regel und erfüllt deren Bedingungen, so passiert – nichts. Wozu das gut sein soll? Immerhin werden der Paket- und der Bytezähler der Regel dadurch inkrementiert, und deshalb kann man diese zu Statistikzwecken oder zur Fehlerverfolgung einsetzen.

7.3.1 Alternative Aktionen

Zusätzlich zu den bereits genannten Aktionen `DROP` und `ACCEPT` gibt es noch weitere, die in Filterregeln auftauchen können. Als wichtigste seien hier genannt:

- REJECT – das Paket zurückweisen
- LOG – das Paket protokollieren

Darüberhinaus können auch eigene Regelketten erzeugt werden, die wie ein Unterprogramm als Aktionen aus anderen Regeln heraus aufgerufen werden (siehe Abschnitt 7.3.4 auf S. 148).

REJECT

Das REJECT unterscheidet sich vom DROP dadurch, dass eine ICMP-Nachricht an die im Paket vorgefundene Absenderadresse gesandt wird, in der in Form einer Codenummer der Grund für die Zurückweisung mitgeteilt wird. Für das betroffene Paket macht dies freilich keinen Unterschied, es haucht sein Netzleben aus, genauso als ob es an einer DROP-Regel gescheitert wäre.

Exkurs

Getreu den RFCs dürfte übrigens kein Paket ohne entsprechende ICMP-Mitteilung an den Absender entsorgt werden, und einige Experten lehnen deshalb die Verwendung von DROP kategorisch ab. Wir vertreten einen pragmatischeren Standpunkt: Nahezu alle Pakete, die von einer sorgfältig konfigurierten Firewall abgewiesen werden, dienen der Vorbereitung oder der Ausführung von Angriffen auf fremde Rechner oder sind schlicht nutzlos. Mit einer Benachrichtigung über ihre Ablehnung würde man daher der Netzgemeinde keinen wertvollen Dienst erweisen – im Gegenteil könnte, wenn die Absenderadresse gefälscht (»gespooft«) ist, ein unbeteiligter Dritter durch die REJECT-Nachrichten sogar gestört werden. Weist man Pakete aus dem eigenen, internen Netz ab, kann REJECT anstelle von DROP die Fehlersuche erleichtern, und es kommt vor, dass durch die Zurückweisung von ident-Anfragen (TCP-Port 113) mittels REJECT Wartezeiten beim Verbindungsaufbau zu einigen externen Servern vermieden werden; in allen anderen Fällen können Sie mit DROP-Regeln und -Policies unnötigen Datenverkehr vermeiden.

Wenn Sie sich beispielsweise direkt ankommender E-Mails pietätvoll per REJECT entledigen wollen, weil Sie Ihre Mail immer direkt vom Provider abholen und zudem die Freude am Spam-Lesen verloren haben, können Sie dies mit einer Regel

```
iptables -A INPUT -p tcp --dport 25 -j REJECT --reject-with icmp-port-unreachable
```

bewerkstelligen. Ein *vorangestelltes*

```
iptables -A INPUT -s 192.168.42.0/24 -p tcp --dport 25 -j ACCEPT
```

lässt Mails aus Ihrem internen Netz (das wir hier als `192.168.42.0/24` annehmen) herein und spiegelt nur externen Absendern vor, dass der SMTP-Port Ihres Rechners unerreichbar wäre.

Andere Rückmeldecodes, die für REJECT in Frage kommen (siehe Ausgabe von `iptables -p icmp --help`), sind unter anderem:

- `--tcp-reset` – TCP-Verbindung durch ein Reset-Paket abbrechen
- `--icmp-host-unreachable` – so tun, als sei der Zielrechner gerade nicht erreichbar
- `--icmp-port-unreachable` – Port ist scheinbar unerreichbar
- `--icmp-admin-prohibited` – der Administrator wünscht keine solche Verbindung

Der letzte Rückmeldecode ist eigentlich der ehrlichste; bei bestimmten Clientprogrammen wird dem Anwender am anderen Ende der Verbindung eine Fehlermeldung präsentiert, die das Wort »verboten« enthält.

LOG

Die LOG-Aktion ist ausgesprochen nützlich: Sie sorgt für eine Protokollierung des Pakets, das die Regel getriggert hat. Alle wichtigen Informationen, insbesondere die IP-Adressen und Portnummern auf der Quell- und der Zielseite, werden in kompakter (und etwas kryptischer) Form in das Syslog-Protokoll geschrieben und stehen für eine spätere Analyse zur Verfügung. Wir raten dringend dazu, alle Pakete, die durch eine DROP- (oder REJECT-) Aktion ins Nirwana befördert werden, mittels einer in der Regelkette unmittelbar vorangehenden LOG-Regel mit genau gleichen Filterkriterien zu protokollieren. Im Gegensatz zu den anderen bisher besprochenen Aktionen hat LOG auf die weitere Verarbeitung der Datenpakete keinen Einfluss.

7.3.2 LOG + DROP: Blockiertes protokollieren

Damit Sie später im Protokoll nicht nur sehen, was für Pakete ausgesondert wurden, sondern auch, welcher Ihrer Regeln (und Überlegungen) sie zum Opfer gefallen sind, ist es sehr ratsam, ein Log-Präfix hinzuzufügen, wie es das Beispiel zeigt (die Option `log--prefix` darf nur hinter dem LOG-Target stehen!). Er wird in den Syslog-Eintrag aufgenommen und sollte in Ihrem Regelwerk eindeutig die Stelle kennzeichnen, an der das Logging stattfand, sodass Sie den Weg des Pakets bis zu seinem Untergang nachvollziehen können.

```
iptables -A FORWARD -p tcp --dport 445 -i eth0 -j LOG --log-prefix "Wurmalarm: "
iptables -A FORWARD -p tcp --dport 445 -i eth0 -j DROP
```

Diese beiden Regeln würden in einem Gateway-Rechner, der ein internes Netz an der Schnittstelle `eth0` mit dem Internet verbindet, alle TCP-Verbindungen zu Port 445 protokollieren und abblocken. Dabei handelt es sich um Windows-Netzwerkzugriffe, die nicht ungeschützt über externe Netze zugelassen werden sollten (nötigenfalls kann man gesicherte Verbindungen über HTTPS oder über VPN verwenden). Hat sich aber ein interner Rechner ein Virus oder einen »Wurm« eingefangen, äußert sich das oft in Tausenden von Versuchen, solche Windows-Verbindungen zu zufälligen IP-Adressen aufzubauen. Durch die Logging-Regel werden diese Versuche im Syslog protokolliert, und ein einfaches `grep` auf das Log-Präfix kann verseuchte Windows-Boxen im eigenen Netz entlarven.

Schon so mancher Firewall-Administrator hat das Logging auch dazu genutzt, seine Filterregeln überhaupt erst vollständig zu konfigurieren: Man beginnt mit einem einfachen, rudimentären Satz von Regeln und schaut immer dann, wenn irgendetwas nicht richtig funktioniert, im Syslog nach, ob zur fraglichen Zeit Logging-Einträge entstanden sind. Wenn ja, entscheidet man anhand der aufgezeichneten Paketdaten, ob der Firewall eine Filterregel hinzugefügt werden muss, die diese Art von Paketen passieren lässt … trotzdem empfehlen wir diese Vorgehensweise ausdrücklich nicht, sondern raten dazu, sich von vornherein einen möglichst vollständigen Regelsatz zu überlegen.

Auch wenn LOG und DROP meist im Doppelpack auftreten, können zur Fehlersuche auch einzelne LOG-Regeln in die Regelketten eingefügt werden, etwa um zu überprüfen, ob bestimmte Pakete, deren Verbleib rätselhaft ist, überhaupt das Firewalling durchlaufen haben. Entfernen Sie aber solche Regeln wieder, wenn sie ihre Schuldigkeit getan haben, denn sonst wird Ihr Syslog unnötig aufgebläht und womöglich in der Folge der Speicherplatz im var- oder root-Dateisystem knapp.

Dies kann freilich auch durch das Loggen der abgewiesenen Pakete passieren, das wir Ihnen ans Herz gelegt haben. Unfreundliche Netzgenossen könnten sogar Ihren Rechner mit Paketen bombardieren, die von Ihrer Firewall abgeblockt werden, nur um Ihr Syslog vollzustopfen und damit Ihren Rechner lahmzulegen – oder auch um das weitere Logging zu unterbinden, damit nachfolgende Hack-Aktivitäten nicht mehr protokolliert werden.

7.3.3 Regeln mit Limit

Als Schutzmaßnahme empfiehlt sich das Rate-Limiting: Mit einer Erweiterung der Netfilter-Regeln kann gezählt werden, wie oft eine bestimmte Regel pro Zeiteinheit getriggert wird, und bei Überschreiten eines festgesetzten Limits kann die Ausführung der mit der Regel verbundenen Aktion unterbunden werden. Wenn Sie beispielsweise eine LOG-Regel auf eine Rate von `6/minute` limitieren, wird (nach einem ersten Schub von fünf Treffern) nur noch höchstens alle 1/6 Minute, also alle 10 Sekunden, die Logging-Aktion einmal ausgeführt. Selbst bei einem massiven Bombardement entstehen so maximal 360 Syslog-Einträge pro Stunde, und dafür wird der Platz auf Ihrer Festplatte wohl ausreichen, bis nachts der `logrotate`-Dämon ein neues Logfile anlegt.

Bei der Limitierung handelt es sich um eine Match-Erweiterung: Wird die betreffende Regel zu häufig aktiviert, meldet die Limit-Routine einfach, dass die Bedingungen der Regel nicht erfüllt seien. Deshalb ist eine Match-Option `-m limit` anzugeben und daran anschließend das gewünschte Limit als Anzahl pro Zeiteinheit, wobei letztere `second`, `minute`, `hour` oder `day` lauten kann (Sekunde, Minute, Stunde oder Tag). Um die Protokollierung im obigen Wurmbeispiel auf einen Eintrag in 10 Sekunden zu beschränken, würden die Kommandos also folgendermaßen ergänzt:

```
iptables -A FORWARD -p tcp --dport 445 -i eth0 -m limit --limit 6/min \
                    -j LOG --log-prefix "Wurmalarm: "
iptables -A FORWARD -p tcp --dport 445 -i eth0 -j DROP
```

Wenn Ihnen der Wert 5 für die Anzahl von Regelaufrufen, die akzeptiert werden, bevor die eigentliche Limitierung einsetzt, nicht zusagt, können Sie mit der Option `--limit-burst` einen anderen Wert festlegen.

7.3.4 Selbst definierte Regelketten

Neben den eingebauten Regelketten, von denen wir bisher INPUT, OUTPUT und FORWARD beschrieben haben, können auch eigene Ketten (»user-defined chains«)

erzeugt werden. Diese werden in der gleichen Weise durch `iptables` mit Regeln bestückt, indem ihr Name im Kommando angegeben wird. Diesen Namen dürfen Sie frei erfinden; um Konflikte mit eingebauten Ketten (auch in Zukunft) zu vermeiden und keine Schwierigkeiten mit der Kommandosyntax zu bekommen, sollte er nur aus Kleinbuchstaben und bei Bedarf Ziffern und Unterstrichen bestehen. Wirksam wird eine solche Kette dadurch, dass ihr Name in einer Regel einer eingebauten Kette als Aktion angegeben wird (`-j eigene_kette`). Dies darf gegebenenfalls an mehreren Stellen geschehen (mit unterschiedlichen Bedingungen verknüpft oder z. B. aus INPUT und FORWARD heraus), allerdings nur in ein und derselben Tabelle (meist `filter`).

Wird für ein Datenpaket als Aktion der Aufruf einer eigenen Regelkette ausgeführt, werden daraufhin deren Regeln der Reihe nach auf das Paket angewendet. Trifft eine von ihnen eine Verfügung über das Paket (etwa ein DROP oder ein ACCEPT), so hat die Filterei ein Ende. Wird hingegen das Ende der selbst gestrickten Kette erreicht, geht es mit der dem Aufruf folgenden Regel weiter, genau wie nach einem Unterprogrammaufruf in einer Programmiersprache. (Selbst definierte Ketten können keine eigenen Policies haben, die nach der letzten Regel wirksam werden.) Auch mehrfach verschachtelte Aufrufe sind möglich, aber kein rekursiver Aufruf derselben Regelkette, denn sonst kämen die Pakete womöglich aus dem Regellabyrinth nicht mehr heraus[10]. Ebenfalls in Analogie zum Programmieren gibt es eine besondere Aktion (`-j RETURN`), mit der eigene Ketten vorzeitig verlassen werden können.

Diese Art von Unterprogrammtechnik dient nicht nur dazu, Informatikern Bewunderung abzuringen, sondern kann bei komplexen Firewall-Konfigurationen wirklich nützlich sein. Anstatt eine Anzahl von gleichartigen Regeln für mehrere Interfaces oder Netzwerke immer wieder neu zu definieren (und bei Bedarf alle parallel abzuändern), kann man sie in einer eigenen Kette zusammenfassen und diese mit unterschiedlichen Interface- oder Adressenbedingungen aufrufen. Weitere Vorteile können sich durch die Verkürzung von Pfadlängen und die Realisierung kompliziert verknüpfter Bedingungen ergeben – wenn es denn unbedingt sein soll. Praxisrelevanter erscheint uns jedoch die Möglichkeit, die Regelstrukturen übersichtlich zu gliedern und änderungsfreundlich zu machen. Das folgende Beispiel soll andeuten, wie die Forwarding-Regeln nach den Richtungen getrennt werden können:

```
iptables -F                     # alle Ketten leeren
iptables -X                     # alle eigenen Ketten entfernen
iptables -P FORWARD DROP        # Forwarding-Policy auf DROP setzen

# eine eigene Kette für Pakete, die ins interne Netz gehen
#
iptables -N in_check
iptables -A in_check -m state --state ESTABLISHED,RELATED -j ACCEPT
```

[10] Es ist sogar möglich, selbst definierte Ketten per »goto« anzuspringen, aber sinnvolle Anwendungen dafür sind ziemlich rar.

```
# ... hier weitere Regeln nach Bedarf einfügen

# eine eigene Kette für Pakete, die nach draußen gehen
#
iptables -N out_check
iptables -A out_check -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A out_check -p udp --dport 53 -j ACCEPT      # DNS-Anfragen zulassen
iptables -A out_check -p tcp --dport 53 -j ACCEPT      # DNS-Anfragen zulassen
iptables -A out_check -p tcp --dport 80 -j ACCEPT      # http zulassen
iptables -A out_check -p tcp --dport 443 -j ACCEPT     # https zulassen
# ... und weitere Regeln für alles, was noch gebraucht wird

# die eigenen Ketten aufrufen
# (eigenes Netz ist an eth0 mit den Adressen 192.168.42.*)
#
iptables -A FORWARD -o eth0 -d 192.168.42.0/24 -j in_check
iptables -A FORWARD -i eth0 -s 192.168.42.0/24 -j out_check

# nun noch INPUT- und OUTPUT-Policies setzen und -Regeln definieren ...
```

In den beiden Ketten `in_check` und `out_check` liegen nun säuberlich getrennt die (meist sehr unterschiedlichen) Regeln für die beiden Richtungen, und spätere Ergänzungen sind sehr leicht durch Anhängen weiterer Regeln an eine der beiden Ketten durchzuführen.

Wie man diese Technik noch verfeinern kann, um das Neuladen von Regeln im laufenden Betrieb zu verbessern, zeigen wir in Abschnitt 7.6 auf S. 160.

7.4 Die wichtigsten Regeln für lokale Firewalls

Wir wollen zunächst die Filterregeln betrachten, die benötigt werden, um einen Linux-Rechner mit Hilfe seiner eigenen Netfilter-Firewall zu schützen. In der Windows-Welt wird eine solche Konfiguration oft als »lokale Firewall« bezeichnet. Im Gegensatz dazu filtert eine netzbasierte Firewall diejenigen Datenpakete, die durch sie hindurchlaufen, und kann damit beispielsweise ein ganzes Subnetz vor Attacken von außen schützen (und möglichst auch umgekehrt den Rest der Welt vor den Aktivitäten von Schadsoftware, die jemand in das Subnetz hineingeschleppt hat).

7.4.1 Filterung ankommender Pakete

Eine lokale Firewall sollte alle Verbindungen abblocken, die von außen aufgenommen werden und nicht an einen Port gerichtet sind, der – vom Administrator beabsichtigt – eine Dienstleistung nach außen hin anbietet. Insofern ist es klar, dass ein System, auf dem z. B. ein Webserver läuft, den TCP-Port 80 allen erwünschten Clients zugänglich machen muss, also Pakete mit jeder beliebigen Absenderadresse akzeptieren muss. Hinzu käme in diesem Fall der TCP-Port 443, über den üblicher-

weise SSL- bzw. TLS-verschlüsselte Webverbindungen abgewickelt werden (meist erkennbar an dem Protokollkürzel `https:` im URL).

Vergessen Sie aber nicht, wenn Sie die INPUT-Regeln für eine lokale Firewall zusammenstellen, dass ein paar Regeln benötigt werden, die nicht zu solchen Serverdiensten gehören, deren Fehlen aber unangenehme Folgen haben kann:

- Pakete, die zu bestehenden Verbindungen gehören oder eine (möglicherweise negative) Antwort auf vorausgegangene Pakete darstellen; in der Netfilter-Terminologie sind das Pakete mit dem Verbindungsstatus ESTABLISHED oder RELATED:

  ```
  iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
  ```

  Damit ist der wichtigste Teil dessen, was hereinkommen soll, schon erfasst, und es sind nur wenige zusätzliche Regeln erforderlich.
- Pakete, die über das Loopback-Netzwerkinterface gesendet wurden; würde man diese abblocken, wäre die Kommunikation zwischen einigen lokalen Programmen gestört:

  ```
  iptables -A INPUT -i lo -j ACCEPT
  ```

- Ping-Pakete (spezielle ICMP-Pakete, die lediglich eine Antwort – ein »Pong« verlangen, um das Funktionieren des Netzwerks zu überprüfen):

  ```
  iptables -A INPUT -p ICMP --icmp-type echo-request -j ACCEPT
  ```

  Nach den Internetkonventionen sollte jeder Host solche Pings beantworten. Da aber in der Vergangenheit Windows-Systeme oft anfällig für bösartig manipulierte Ping-Pakete waren und auf sie zumindest mit Systemabstürzen reagierten (»ping of death«), glauben viele Computerbesitzer, sie wären besonders gut geschützt, wenn ihr Rechner nicht auf Ping antwortet. Sie hoffen zudem, dass die bösen Buben solchermaßen »geschützte« Systeme gar nicht finden werden. Letzteres stimmt leider nicht – es gibt genügend andere Methoden, um Rechner zu entdecken. Linux hat sich noch nie zu Tode pingen lassen, und zudem ist das Ping ein wertvolles Mittel, um das Funktionieren des Netzes zu testen.

  Deshalb wollen wir hier ein gutes Wort für das Ping einlegen und plädieren dafür, es zumindest für alle Adressen aus dem lokalen Subnetz, besser aber generell freizugeben (vielleicht mit einer zusätzlichen Ratenbegrenzung):

  ```
  iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
  #
  # oder z.B. mit Beschränkung auf das lokale Netzwerk, hier 192.168.42.*:
  # iptables -A INPUT -p icmp --icmp-type echo-request -s 192.168.42.0/24 -j ACCEPT
  ```

- Ident-Anfragen: Diese waren früher mal sinnvoll, als eine große Zahl von Benutzern gemeinsam einen Unix-Host benutzte. Bei einer Anfrage an einen externen Server konnte dieser mittels Ident herausfinden, von welcher Benutzerkennung

die Anfrage ausging. Manche (z. B. IRC-) Server glauben noch heute, wo doch die meisten Internet-Nutzer ihren eigenen Rechner administrieren, dass Ident-Anfragen brauchbare Antworten liefern würden.

Die einfachste Methode, auf diese Anfragen zu reagieren und unnötige Wartezeiten zu vermeiden, in denen der Server auf Antwort hofft, besteht darin, sie postwendend abzulehnen:

```
iptables -A INPUT -p tcp --dport 113 -j REJECT
```

Zusammengefasst ergibt sich daraus die folgende Kommandosequenz, die je nach Bedarf noch zu ergänzen ist:

```
iptables -P INPUT DROP                  # Policy auf DROP setzen
iptables -F INPUT                       # etwaige vorhandene Regeln löschen
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -i lo -j ACCEPT       # rechnerlokale Verbindungen zulassen
iptables -A INPUT -p ICMP --icmp-type echo-request \
                 -m limit --limit 3/sec -j ACCEPT  # Ping begrenzt erlauben
iptables -A INPUT -p tcp --dport 113 -j REJECT
iptables -A INPUT -m limit --limit 6/min -j LOG --log-prefix "input blocked: "
```

7.4.2 Filterung ausgehender Pakete

Während es ganz offenkundig ist, dass alle ankommenden Verbindungen abgeblockt werden sollten, die nicht auf einen bewusst angebotenen Netzwerkdienst zugreifen, gibt es bezüglich der Filterung abgehender Verbindungen unterschiedliche Strategien. Die erste besteht darin, alle vom lokalen Rechner aus initiierten Verbindungen zuzulassen – in der Hoffnung, dass sie von einem berechtigten Nutzer veranlasst wurden und einen sinnvollen Zweck erfüllen. Hierzu kann für die Policy der OUTPUT-Regelkette der Wert `ACCEPT` eingestellt werden, der ohnehin nach dem Systemstart gültig ist, und auf weitere OUTPUT-Regeln verzichtet werden. Mit dieser Vorgabe ergeben sich für lokale Nutzer praktisch keine Einschränkungen bei der Nutzung des Netzwerks, also eine maximale Bequemlichkeit, weswegen sie auch für Workstations und normale PCs gern angewendet wird.

Die Alternative besteht darin, alle abgehenden Verbindungen grundsätzlich abzulehnen und durch eine Reihe gezielter Filterregeln lediglich diejenigen zuzulassen, von denen man weiß, dass sie tatsächlich benötigt werden. Diese Strategie bietet sich insbesondere auf Serversystemen an, auf denen ohnehin keine lokalen Benutzer routinemäßig arbeiten (sollten), die an solch einer restriktiven Einstellung womöglich verzweifeln würden.

Die Möglichkeit, dass auch ein gut abgesichertes Serversystem durch eine Schwachstelle in einem der nach außen hin angebotenen Dienste unter fremde Kontrolle gerät, ist nie ganz auszuschließen. Eine strikte Output-Filterung hindert Schadprogramme, die den Weg ins System gefunden haben, daran, andere Rechner im Intra- oder Internet anzugreifen oder die erfolgreiche Infektion an einen Hacker zu mel-

den und auf dessen weitere Instruktionen zu warten, jedenfalls solange die bösartige Software ohne root-Rechte abläuft und somit nicht die Firewall abschalten kann.

Gewisse Verbindungen muss allerdings auch ein solchermaßen abgesicherter Server nach außen aufnehmen können: Er sollte

- DNS-Anfragen zumindest an den zuständigen Nameserver richten können (bzw. an beliebige andere Rechner, wenn er selbst DNS-Server ist),
- Pings aussenden können, damit Netzwerkprobleme einfach diagnostiziert werden können, und
- Software-Updates abrufen können.

Von welchem Server und über welches Protokoll die Updates geholt werden, können Sie bei bei Debian-Systemen in der Datei `/etc/apt/sources.list` nachlesen und unter SUSE im YaST-Menü zum Online-Update sehen. Wenn beispielsweise die `sources.list` den Eintrag

```
deb ftp://ftp.de.debian.org/debian stable main contrib non-free
```

enthält, müssen Sie FTP-Verbindungen zu `ftp.de.debian.org` zulassen. Achten Sie darauf, dass das Modul `ip_conntrack_ftp` geladen ist, und bestimmen Sie die IP-Adresse des FTP-Servers:

```
hugo:~# host ftp.de.debian.org
ftp.de.debian.org has address 141.76.2.4
hugo:~#
```

Es ist zwar möglich, aber nicht sinnvoll, Rechnernamen in iptables-Regeln einzutragen, da das Erzeugen dieser Regeln mit einer Fehlermeldung (und nach einiger Wartezeit) abgewiesen würde, wenn die Namensauflösung zufällig einmal nicht möglich wäre. Oft werden die Firewall-Regeln sogar schon eingerichtet, bevor überhapt das Netzwerk aktiviert ist, und damit würde die Namensauflösung immer schiefgehen. Allerdings besteht bei Verwendung der numerischen IP-Adressen das Risiko, dass der externe Server irgendwann einmal (unter Beibehaltung seines Namens) auf eine andere Adresse umzieht, was an dieser Stelle bedeuten würde, dass die Updates nicht mehr funktionieren, bis der Administrator dies bemerkt und die Firewall-Regel abändert.

Dieses Problem können Sie, wenn Sie ein Netz aus mehreren Linux-Rechnern administrieren, dadurch entschärfen, dass Sie die benutzte Distribution einschließlich der aktuellen Updates auf einem einzelnen Rechner spiegeln und die anderen so konfigurieren, dass sie die Updates und neu zu installierende Pakete von diesem Spiegel holen. Für die Debian-Distribution gibt es ein Paket `apt-proxy`, das ein lokales Paket-Repository automatisch verwaltet und aktualisiert sowie die für diese Distribution notwendigen Verzeichnisdateien passend erzeugt; die anderen lokalen Rechner können in ihrer `sources.list`-Datei mit einem Eintrag

```
deb http://debproxy.beispiel.org:9999/debian stable main contrib non-free
```

auf den APT-Proxy `debproxy.beispiel.org` verweisen und in ihren Firewall-Regeln den Zugang zu dessen (numerischer) IP-Adresse unter TCP-Port 9999 erlauben.

Fassen wir noch einmal die Kommandos zusammen, mit denen eine restriktive Output-Policy umgesetzt werden kann:

```
# Grundsätzlich alles verwerfen, was nicht erwünscht ist
iptables -P OUTPUT DROP

# Und jetzt kommen die Ausnahmen ...

# Ausgehende Pakete erlauben, die zu bestehenden Verbindungen gehören
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Ping erlauben
iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
# Beispiele für den Zugriff auf neue Pakete und Updates:

# Debian-Distribution mit externen Repositories

# FTP-Zugang zu ftp.de.debian.org
modprobe ip_conntrack_ftp
iptables -A OUTPUT -p tcp --dport 21 -d 141.76.2.4 -j ACCEPT
# HTTP-Zugang zu security.debian.org - dahinter stecken mehrere IPs
for ip in 128.101.80.133 194.109.137.218 82.94.249.158
do
        iptables -A OUTPUT -p tcp --dport 80 -d $ip -j ACCEPT
done

# Oder lediglich Zugang zu einem apt-proxy, z.B. auf 192.168.3.2:
# iptables -A OUTPUT -p tcp --dport 9999 -d 192.168.3.2
```

Falls Sie aber – entgegen aller Planung und allen guten Vorsätzen – doch einmal auf dem solchermaßen gut geschützten Server arbeiten und vielleicht akute Fehler suchen müssen, ist es zugegebenermaßen ausgesprochen lästig, trotz bestehender Internet-Anbindung nicht auf externe Websites, FTP- und E-Mail-Server zugreifen zu können. Sollten Sie in dieser Situation einen anderen PC in Reichweite haben, der weniger kritisch und deswegen liberaler konfiguriert ist, würden wir dazu raten, diesen zu nutzen und heruntergeladene Dateien über eine ssh-Verbindung auf den Server zu übertragen; steht dieser Umweg jedoch nicht zur Verfügung, bleibt Ihnen nichts anderes übrig, als die hochsichere Server-Konfiguration für die Dauer der Wartungsarbeiten auf gewöhnliches PC-Niveau abzusenken, indem Sie mit der Eingabe

```
hugo:~# iptables -I OUTPUT 1 -j ACCEPT
hugo:~#
```

jedweden ausgehenden Datenverkehr zulassen. Natürlich könnten Sie stattdessen auch die Output-Policy auf `ACCEPT` setzen, aber dann würden Ihre Output-Firewall-Regeln dennoch für jedes Paket durchlaufen, das den Rechner verlässt, und wenn DROP- oder REJECT-Regeln darunter sind, gehen weiterhin Pakete verloren.

Vergessen Sie nicht, das Kommando

```
hugo:~# iptables -D OUTPUT -j ACCEPT
hugo:~#
```

einzugeben, sobald die Situation sich normalisiert hat und der Server wieder optimal geschützt werden kann.

7.5 Linux als Router

Wenn Sie einen Linux-Rechner mit eingeschaltetem IP-Forwarding als Router einsetzen, tun Sie immer gut daran, den Durchgangsverkehr mit einem möglichst restriktiven Satz von FORWARD-Regeln auf das Nötigste zu beschränken. Das wird schwer sein, wenn es im internen Netz einen Power-User gibt (außer Ihnen selbst), der sich nicht damit abfinden mag, dass das Firewalling seinem Tatendrang Grenzen setzt und er (oder sie) nicht ungehindert auf alle Multimedia-, Spiele- und ominösen Webangebote auf Nichtstandardports zugreifen kann. Vielleicht hilft es, so einen Problem-User in ein separates Subnetz auszulagern, das zum Internet hin permissiv, zum übrigen internen Netz aber so restriktiv wie möglich gefiltert wird.

Ansonsten gelten im Grunde die gleichen Überlegungen wie für lokale Firewalls: Hereinkommen (aus dem Internet oder allgemein vom offeneren in das geschlossene, Ihnen mehr am Herzen liegende Netz) lässt man möglichst gar nichts außer den Paketen mit dem Verbindungsstatus `ESTABLISHED` oder `RELATED`, und hinaus nur das wirklich Benötigte. Wenn Sie wie beschrieben die Forwarding-Regeln in separate Ketten je Richtung, `in_check` und `out_check`, einsortieren, könnten Sie für die erstere mit

```
iptables -A in_check -m state --state ESTABLISHED,RELATED -j ACCEPT
```

plus je eine oder mehreren Regeln für jeden Server, der von außen zu erreichen sein soll, wie beispielsweise

```
iptables -A in_check -p tcp --dport 80 -j ACCEPT     # http
iptables -A in_check -p tcp --dport 443 -j ACCEPT    # https
```

für einen normalen Webserver, schon die Aufgabe erledigt haben. Für die nach draußen gehende Richtung sollten natürlich auch die durch ESTABLISHED und RELATED erfassten Pakete akzeptiert werden. Hinzu kommen normalerweise:

- Webzugriffe (wer will darauf schon verzichten?) zu den TCP-Ports 80 und 443, wenn nicht die Nutzung eines Web-Proxy erzwungen wird.

- Ping zu fremden Rechnern (ICMP-Typ `echo-request`) muss nicht unbedingt allgemein freigegeben werden, denn viele Anwender kennen es nicht, und manche Schadsoftware sendet massenhaft Pings aus; für erfahrene User kann es allerdings ein nützliches Werkzeug sein.
- DNS-Zugriffe (Ports 53/udp und 53/tcp), sofern diese nicht über einen internen DNS-Server abgewickelt werden.
- FTP-Zugriffe (Port 21/tcp), weil in vielen Download-URLs FTP als Protokoll spezifiziert ist.
- NTP-Zugriffe (Port 123/udp und 123/tcp) zur Synchronisierung von Rechneruhren durch funkuhrgesteuerte Zeitserver; dies kann entfallen, wenn es im internen Netz einen Zeitserver gibt.
- Abholen von E-Mails von externen Mailservern (per IMAP, TCP-Ports 143 und 993 ohne bzw. mit SSL-Verschlüsselung, notfalls auch POP3, TCP-Ports 110 und 995).
- Der Versand von E-Mails (SMTP, TCP-Port 25) sollte nur einem regulären Mailserver erlaubt sein; was ein normaler PC über SMTP hinausschickt, sind praktisch immer nur Spam und Viren.
- SSH-Verbindungen (TCP-Port 22), sofern die internen User überhaupt wissen, was man damit macht, und verantwortungsvoll damit umgehen (man kann damit gewissermaßen Löcher in die Firewall bohren).
- Windows-spezifische Verbindungen (vor allem über die Ports 135, 137, 139 und 445) stellen ein grandioses und vielfach erprobtes Einfallstor für Schädlinge dar und sollten erstens gar nicht, zweitens nur, wenn es nicht anders geht, und drittens nur mit genauer Festlegung der beteiligten Quell- und Zieladressen freigegeben werden!

Weitere individuell benötigte Portfreigaben können nach Bedarf hinzukommen, gegebenenfalls per `-s`-Option auf eine Nutzung durch bestimmte interne IP-Adressen beschränkt. Je ausgefallener eine Anwendung ist, umso schwieriger kann die Beschaffung von Informationen über benötigte Ports werden, und Auskünfte der Art, für die Anwendung seien in der Firewall alle Ports von 1024 bis 32767 freizugeben, sind nicht sehr hilfreich (und meist auch unwahr). Notfalls müssen Sie die notwendigen Regeln doch per »trial and error« ermitteln …

7.5.1 Network Address Translation (NAT)

Bei der Network Address Translation (NAT) geht es um die Manipulation von Paketen an der Grenze von Netzwerkbereichen. Es sind verschiedenste NAT-Szenarien denk- und realisierbar – meist geht es allerdings darum, innerhalb eines internen (häuslichen oder Firmen-) Netzes private Adressen zu verwenden, weil man beispielsweise keine offiziellen, extern nutzbaren IP-Adressen für alle seine Rechner, Drucker etc. zur Verfügung hat. Für solche Zwecke sind die im Internet nicht gerouteten, reservierten Adressbereiche

- 10.0.0.0/8 (10.0.0.0 – 10.255.255.255)

- 172.16.0.0/12 (172.16.0.0 – 172.31.255.255)
- 192.168.0.0/16 (192.168.0.0 – 192.168.255.255)

geeignet. Gern verwendet man die 192.168-Adressen mit einer beliebigen Zahl von 1 bis 254 an dritter Stelle, weil diese von vielen Programmen als 24-Bit-Subnetze erkannt werden, wie beispielsweise das in diesem Buch oft strapazierte Netz 192.168.42.0/24.

Nach außen sollen solche mit privaten Adressen bedachten Stationen nur unter einer gemeinsamen, nämlich der einzigen verfügbaren »echten« IP-Adresse auftreten. Dazu müssen an der Übergangsstelle vom internen zum externen Netz, dem Gateway-Router, alle nach draußen gehenden Pakete umgeschrieben und die Absenderadressen durch die offizielle Adresse ersetzt werden – was eigentlich kein Problem darstellt –, und in den hereinkommenden Paketen muss umgekehrt die immer gleiche Zieladresse durch die jeweils richtige interne Empfängeradresse ersetzt werden. Dies ist sehr wohl ein Problem, weil dazu bekannt sein muss, an welche Station im internen Netz das Paket denn wohl gehen soll.

Lösbar ist dieses Problem für Antwortpakete zu Anfragen, die von einem internen Rechner gekommen sind: Man braucht sich nur zu merken, von welcher IP-Adresse und Portnummer die Anfrage kam und an wen sie ging. Kommt die passende Antwort von dort zurück, kennt man den richtigen Adressaten und kann sie weiterleiten. Allerdings kann es sehr wohl passieren, dass zwei interne Rechner zur gleichen Zeit Anfragen an das gleiche Ziel richten (z. B. an eine bekannte Suchmaschine oder an den Webserver unseres Verlegers …). Wie hält man die dazugehörenden Antworten auseinander?

Das gelingt mit einem Trick: Das Gateway, das die Adressumsetzung vornimmt, vergibt für jede der Anfragen eine eigene, neue Absenderportnummer. Diese Nummern werden sowieso senderseitig meist (mehr oder weniger) zufällig bestimmt, und der Empfänger interessiert sich nicht weiter dafür, als dass er sie in die Antworten (sozusagen als »zu Händen«-Vermerk) wieder hineinschreibt. Das Gateway kann also anhand der zielseitigen Portnummer die Antworten zuordnen und an den wahren Empfänger weiterleiten. Ein Beispiel: Die Anfragen

- von `192.168.42.5`, Port 12345, an `134.99.128.40`, Port 80
- von `192.168.42.7`, Port 23456, an `134.99.128.40`, Port 80

werden vom Gateway mit der (momentanen) Adresse `1.2.3.4` ins Internet gesendet mit neu ausgedachten Portnummern als Anfragen

- von `1.2.3.4`, Port 14711, an `134.99.128.40`, Port 80
- von `1.2.3.4`, Port 14712, an `134.99.128.40`, Port 80

Kommt nun ein Paket herein

- von `134.99.128.40`, Port 80, an `1.2.3.4`, Port 14712

so gehört es zur zweiten Anfrage und wird ins interne Netz weitergeleitet als Paket

- von `134.99.128.40`, Port 80 an `192.168.42.7`, Port 23456

– und für den Empfänger ist die Welt in Ordnung: Von der ganzen Umschreiberei bekommt er nichts mit. Er glaubt, er habe seine Antwort direkt vom Server erhalten.

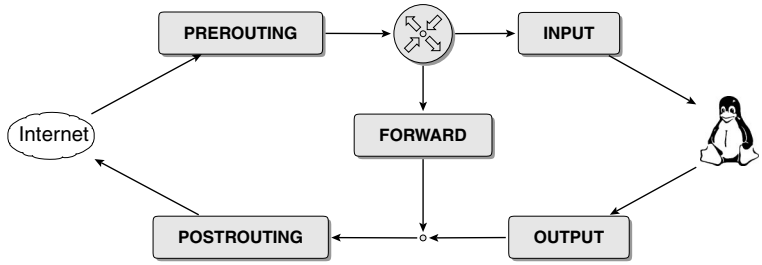


Abbildung 7.3: Paketfluss mit PREROUTING- und POSTROUTING-Ketten

Ein kleiner Wermutstropfen: Leider lassen sich nicht alle Anwendungsprotokolle unbemerkt »natten«. Für einige (wie FTP) bieten die Helper-Module des Connection Tracking Abhilfe (siehe S. 130). Andere, oft etwas exotische oder angestaubte Protokolle funktionieren über NAT definitiv nicht.

Damit ein als Router werkelndes Linux-System in der beschriebenen Weise NAT betreiben kann, muss es alle Pakete, die zwischen den Netzwerken hin- und hergehen, anhand einer Zuordnungstabelle umschreiben. In der Hauptsache ist dafür der Netfilter-Code zuständig, denn diese Aufgabe lässt sich sehr schön in Verbindung mit dem Connection Tracking erledigen. Die zugehörigen Aktivitäten sind im Linux-internen Paket-Straßennetz ganz weit in Richtung Netzwerk verlagert, weil es sich bei den manipulierten Adressen quasi um reine Äußerlichkeiten handelt: Eine besondere Regelkette namens POSTROUTING ist für das Umschreiben der Absenderadressen unmittelbar vor dem Absenden der Daten zuständig, und als Gegenstück korrigiert die Kette PREROUTING die Zieladressen sofort nach dem Empfang.

Die ähnlich lautenden Namen dieser Ketten stiften oft Verwirrung. Man kann sich aber gut merken, dass sich das interne *Routing* immer mit den *Zieladressen* beschäftigt und diese somit vorher, also *pre-routing*, passend gesetzt sein müssen. Die Absenderadressen dagegen sind für das Routing irrelevant und werden erst *post-routing* gefälscht.

7.5.2 Source-NAT (SNAT)

Am besten betrachtet man bei der Konfiguration von NAT immer nur das erste Paket einer Verbindung. In unserem Szenario ändert es seinen Absender (die »source address«), was im Postrouting passieren muss, und man spricht von Source-NAT (SNAT). Dass die Antworten richtig behandelt werden, kann man getrost voraussetzen.

Aktiviert wird SNAT durch Regeln in der POSTROUTING-Kette, die die Aktion `SNAT` enthalten. Sie dürfen nur in der Tabelle `nat` stehen – in der sonst üblicherweise

benutzten `filter`-Tabelle gibt es diese Kette gar nicht. Die Option `-t nat` muss also unbedingt angegeben werden. Außerdem ist die IP-Adresse festzulegen, die als Absenderangabe in die hinausgesendeten Pakete einzutragen ist. Für unser Beispiel kann das Kommando somit lauten:

```
iptables -t nat -A POSTROUTING -s 192.168.42.0/24 -o eth1 -j SNAT --to 1.2.3.4
```

Mit vollem Namen heißt die SNAT-Option `--to-source`, und man kann auch einen Adressenbereich sowie die zu benutzenden Ports angeben. Beides ist aber im Normalfall unnötig.

Hier wurde durch `-s` festgelegt, dass nur diejenigen Pakete »genattet« werden sollen, die aus dem internen Netz kommen, und durch `-o` wird das SNAT auf Pakete eingeschränkt, die das Gateway in Richtung Außenwelt verlassen (die sei hier durch `eth1` erreichbar). Es empfiehlt sich grundsätzlich, genau zu sagen, auf welche Pakete das NAT wirken soll, denn wenn die falschen Pakete davon betroffen werden, treten schwer zu lokalisierende Fehler auf.

Wenn das Gateway sich (per Modem oder DSL) ins Internet einwählt, hat diese Lösung zwei Schönheitsfehler: Erstens ist die SNAT-Adresse nicht fest, sondern ändert sich bei jeder Einwahl bzw. als Schikane seitens des Providers spätestens nach 24 Stunden, und zweitens enthalten die Netfilter-Tabellen nach einer Änderung der externen Adresse noch eine Menge Connection-Tracking- und NAT-Einträge mit der alten Adresse, die zumindest nutzlos und womöglich sogar auch mal schädlich sind. Deshalb ist für solche dynamischen SNAT-Adressen die Aktion `MASQUERADE` geschaffen worden, die keine Adressangabe benötigt, sondern einfach die Adresse des ausgehenden Interfaces benutzt. Und wenn dieses Interface beim Adresswechsel zumindest kurzzeitig inaktiv wird, sorgt `MASQUERADE` für die Löschung der nicht mehr relevanten Tracking-Einträge. Bei einer Einwahlverbindung sollte das Kommando also so aussehen:

```
iptables -t nat -A POSTROUTING -s 192.168.42.0/24 -o eth1 -j MASQUERADE
```

7.5.3 Destination-NAT (DNAT)

Private Adressen stellen für die Rechner im internen Netz einen gewissen Schutz dar, weil sie von außen nicht erreicht werden können und sogar ihre schiere Existenz vor der Außenwelt weitgehend verborgen bleiben kann. Vor allem Windows-Rechner danken dies durch deutlich verringerte Infektionsanfälligkeit. Andererseits erzielt eine restriktive Filterung im Gateway (oder, bei Linux, im Rechner selbst) im Prinzip auch ohne NAT den gleichen Grad an Sicherheit.

Gibt es aber im internen Netz einen Webserver, der doch von außen erreichbar sein soll, stehen wir vor einem neuen Problem. Direkt ansprechbar ist er von externen Clients nicht, weil ja die internen Adressen außerhalb unseres Hoheitsbereichs gar nicht sichtbar sind.

Die einzige Möglichkeit des Kontakts besteht darin, von außen die Adresse des Gateways anzusprechen und von diesem die ankommenden Pakete ins interne Netz weiterleiten zu lassen. Da bei der Aufnahme einer solchen Verbindung die Zieladresse (die »destination address«) manipuliert werden muss, handelt es sich um Destination-NAT (DNAT), das im Prerouting durchzuführen ist.

Die ankommenden Pakete sind zunächst ans Gateway selbst gerichtet; dass sie für jemand anderen bestimmt sind, kann nur anhand der Zielportnummer ermittelt werden: Wenn das Gateway selbst keinen Webserver auf Port 80 betreibt (und als Firewall und Router sollte es das auch besser nicht), kann man festlegen, dass Verbindungen an diesen Port an den internen Webserver weiterzureichen sind, beispielsweise an die Adresse `192.168.42.5` und ebenfalls an Port 80. Da hierbei die Portnummer darüber entscheidet, ob ein Forwarding stattfindet oder nicht, nennt man diese Methode auch »Port Forwarding«. Im `iptables`-Kommando ist dafür die Aktion DNAT zuständig:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to 192.168.42.5
```

Problematisch wird es, wenn ein weiterer Webserver aus dem internen Netz nach außen sichtbar werden soll, denn der mit der Gateway-Adresse verbundene Port 80 ist ja bereits mit einem DNAT belegt. Also muss man sich eine andere Portnummer ausdenken – zum Beispiel 88 – und diese auf den Port 80 des zweiten Servers »dnatten«:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to 192.168.42.5
```

Natürlich müssen die externen Nutzer diesen Server dann unter der abweichenden Portnummer ansprechen, d. h. in unserem Fall, die Adresse `http://1.2.3.4:88/` in ihren Browser eintippen.

Der zu Beginn dieses Abschnitts diskutierte Schutz interner Rechner durch NAT wird zwar etwas löcherig, wenn einzelne von ihnen per DNAT erreichbar werden, aber immerhin wird nur ein bestimmter Service auf einem einzelnen Port exponiert. Das ist zumindest besser, als wenn der Server direkt und ungeschützt am Internet hinge.

7.6 Tipps zur Erstellung von Firewall-Prozeduren

Die bisher gezeigten Beispielprozeduren haben den Nachteil, dass beim Neuladen der Firewall-Regeln für eine kurze Zeit sämtliche eingehenden Pakete verworfen werden: Durch `iptables -F INPUT` werden alle bisherigen INPUT-Regeln gelöscht, und da die INPUT-Policy mit `iptables -P INPUT DROP` schon beim ersten Durchlaufen der Prozedur auf den Wert `DROP` gesetzt wurde, kommt überhaupt kein Paket mehr durch.

Üblicherweise ist dieser Effekt nicht tragisch; ein paar einzelne Datenpakete können immer einmal verloren gehen, und bei TCP-Verbindungen werden die betroffenen

Pakete sogar, wenn die Unterbrechung nicht zu lange dauert, automatisch wiederholt. Wenn aber Ihr Linux-System für ein ganzes Netzwerk als Firewall und Router arbeitet und womöglich durch einen Tippfehler in der Prozedur die Forwarding-Regeln nicht sogleich wieder ordentlich aufgebaut werden, ist das lokale Netz praktisch vom Internet getrennt. Die betroffenen Anwender nutzen erfahrungsgemäß die gewonnene Zeit dazu, dem Administrator die Hölle heiß zu machen.

Würde man hingegen (zum Schutz des Administrators) vor dem Neuladen der Regeln die Policy auf `ACCEPT` setzen, so käme es im Fehlerfall nicht zu einer Unterbrechung; stattdessen würde aber der Firewall-Schutz des internen Netzes völlig aufgehoben, was vielleicht die noch schlechtere Alternative wäre.

Es gibt einen Trick, den fortgeschrittene Firewall-Administratoren anwenden können, um die Regeln praktisch unterbrechungsfrei neu laden zu können. Wir wollen ihn hier anhand einer Beispielprozedur demonstrieren:

Listing 7.1: /etc/init.d/myfirewall

```
#! /bin/sh

set -e

PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
DESC="load firewall rules"
NAME="myfirewall"
SCRIPTNAME=/etc/init.d/$NAME

function set_input_rules ()
        if iptables -L in_check -n >/dev/null 2>&1
        then
                # Wenn eine Regelkette dieses Namens schon existierte:
                iptables -E in_check in_check_tmp   # umbenennen
        fi

        if [ $1 != "stop“ ]
        then
                # Wenn das Firewalling nicht gestoppt werden soll:
                iptables -N in_check               # neue Regelkette anlegen
                iptables -A in_check -m state
                         --state ESTABLISHED,RELATED -j ACCEPT
                iptables -A in_check -i lo -j ACCEPT
                iptables -A in_check -p icmp --icmp-type echo-request -j ACCEPT
                #
                # Hier nach Bedarf die weiteren Regeln eintragen!
                #
                iptables -A INPUT -j in_check       # aufrufen lassen
        fi
```

```
        if iptables -L in_check_tmp -n >/dev/null 2>&1
        then
                # Wenn eine umbenannte Regelkette existiert:
                iptables -D INPUT -j in_check_tmp   # ihren Aufruf löschen
                iptables -F in_check_tmp            # leerputzen
                iptables -X in_check_tmp            # entsorgen
        fi


case "$1" in
  start|reload|restart|force-reload)
        echo -n "Starting $DESC: $NAME"
        set_input_rules start
        echo "."
        ;;
  stop)
        echo -n "Stopping $DESC: $NAME"
        set_input_rules stop
        echo "."
        ;;
  *)
        echo "Usage: $SCRIPTNAME start|stop|restart|reload|force-reload" >&2
        exit 1
        ;;
esac

exit 0
```

Um das Beispiel nicht zu umfangreich werden zu lassen, haben wir hier nur INPUT-Regeln behandelt. Für OUTPUT und gegebenenfalls FORWARD sowie die NAT-Ketten kann man analoge Funktionen `set_output_rules` etc. definieren und zum Starten und Stoppen aufrufen.

Diese Lösung macht sich die Möglichkeit zunutze, mit `iptables -N` eigene Regelketten anzulegen. Hier wird eine solche Kette namens `in_check` angelegt, die in gleicher Weise mit Regeln gefüllt wird (hier durch `iptables -A in_check ...`) wie sonst INPUT, FORWARD oder OUTPUT. Sie kann ähnlich wie ein Unterprogramm aufgerufen werden, indem ihr Name als Ziel in einer Regel spezifiziert wird (mit der Option `-j`). Nach Abarbeitung der in der Kette enthaltenen Regeln kehrt die Prüfung eines Pakets wieder zur Aufrufstelle zurück und fährt dort mit den nächsten Regeln fort, solange nicht über das Schicksal des Pakets durch eine Aktion wie ACCEPT oder DROP entschieden wurde.

Eine selbst definierte Regelkette kann umbenannt werden; davon wird hier Gebrauch gemacht und einer womöglich existierenden Kette `in_check` der Name `in_check_tmp` gegeben, bis die neuen Regeln eingetragen und aktiviert worden sind.

Danach wird sie entsorgt: Der Aufruf wird entfernt, sodann werden alle ihre Regeln verworfen (durch `iptables -F`), und schließlich wird sie gelöscht (`iptables -X`). Fantasiebegabte Administratoren werden dieses Schema noch weiter vervollkommnen wollen; für die Firewall eines gewöhnlichen Heimrechners grenzt jedoch diese Prozedur schon an Overkill.

8 Komfortabel E-Mail speichern und verwalten

E-Mail ist neben dem Web der Service, der das Internet zu dem Kommunikationsmedium gemacht hat, das es heute ist. Außerdem gehören die E-Mail-Protokolle zu den ersten und ältesten, die noch für den Vorgänger des Internet, das ARPANet, entwickelt wurden. Der Verarbeitung lagen Gegebenheiten zugrunde, die sich von den heutigen stark unterscheiden: Es gab noch keine PCs, alles wurde auf Servern abgewickelt, die i. d. R. auch rund um die Uhr liefen. Das hatte zur Folge, dass die serverseitige Abwicklung der E-Mail-Übermittlung davon ausging, dass die empfangenden Rechner jederzeit zu erreichen waren[1].

Der Missbrauch der E-Mail durch die sogenannten Spam-Mails hat dazu geführt, dass die Mailserver mit Zusatzfunktionen versehen wurden, um die ungewollte Weiterleitung von E-Mails zu verhindern. Um die Problematik zu verstehen und für die jeweilige Situation den passenden Lösungsansatz zu finden, können wir auf ein paar theoretische Grundlagen nicht verzichten.

8.1 Grundlagen der Mailübermittlung

Eigentlich ist doch alles ganz einfach: Wenn wir eine E-Mail verschicken, rufen wir ein Programm auf, mit dem wir sie schreiben und versenden. Um E-Mail *bekommen* zu können, braucht man aber noch »eine Mailadresse«. Und an dieser Stelle wird es den meisten Mail-Nutzern auf Grund der Komplexität der Abläufe dann zu kompliziert. Man merkt sich i. d. R. nur, dass man irgendwo »Daten des Providers« eingetragen hat und die E-Mail dann bei einem ankommt. Leider reicht dieses Wissen nicht aus, um einen Mailserver zu *betreiben*; daher wollen wir uns den Weg einer E-Mail einmal näher ansehen:

- Ein Absender schreibt eine E-Mail in einem Mailprogramm seiner Wahl und verschickt sie. Das Mailprogramm liefert diese E-Mail jetzt bei einem Mailser-

[1] Es gab damals natürlich auch Rechner, die sich per Modem in das Netzwerk einwählten. Dazu wurde ein spezieller Dienst entwickelt, der heute aber kaum noch genutzt wird: UUCP erlaubte es, die eigentlich synchron ablaufenden Dienste auf asynchrone Modem-Verbindungen abzubilden. Dieser Dienst unterscheidet sich jedoch fundamental von den heutzutage für die Einwahl genutzten Methoden.

ver ab. Welcher das ist, hängt von der Arbeitsweise des Programms ab. Windows-Programme versuchen häufig, die E-Mail direkt beim zuständigen Server abzuliefern. Linux-Programme sind i. d. R. so eingestellt, dass sie einem lokal installierten Server die Auslieferung überlassen.

- Der Mailserver versucht nun, dem Rechner die E-Mail zu übergeben, der für die als Empfänger eingetragene Adresse zuständig ist. Dazu muss er über das Domain Name System herausbekommen, welcher **Mail-Exchanger** für den Domain-Namen der Adresse festgelegt ist. Er wertet zuerst den sogenannten *MX-Record* für die Absenderadresse aus. Erhält er auf diese Art und Weise keine IP-Adresse, versucht er, die E-Mail an den Rechner auszuliefern, dessen Name oder IP-Adresse in der Empfängeradresse angegeben ist.
- Auf dem empfangenden Server muss jetzt ein Prozess laufen, der auch während der Abwesenheit des eigentlichen Empfängers die E-Mail zwischenspeichert. Kann der versendende Server die E-Mail nicht abliefern, versucht er es zwar in regelmäßigen, größer werdenden Abständen erneut; die meisten Mailserver schicken eine nicht auslieferbare E-Mail aber nach einer festgelegten Anzahl von Versuchen mit dem Vermerk »unzustellbar« zurück. Der Mailserver Exim4 z. B. ist bei der Installation so konfiguriert, dass er eine E-Mail innerhalb der ersten zwei Stunden alle 15 Minuten auszuliefern versucht, dann 16 Stunden lang in immer länger werdenden Abständen beginnend mit einer Stunde und danach bis zu vier Tage lang alle 6 Stunden. Kann die E-Mail nach vier Tagen immer noch nicht ausgeliefert werden, schickt er sie mit einer Fehlermeldung an den Absender zurück.

Achtung

Die Mechanismen, die dafür sorgen, dass eine E-Mail auch ankommt, wenn der Ziel-Server aktuell nicht erreichbar ist, sind nicht auf eine unregelmäßige Einwahl ausgelegt.

- Wenn Sie sich auf einem Linux-Rechner auf der Textkonsole anmelden, werden Sie beim Einloggen darüber informiert, dass auf dem System E-Mails für Sie vorhanden sind. Wenn Sie sich auf dem Rechner angemeldet haben, auf dem der Mailserver auch die eingehenden Mails abspeichert, finden Sie diese i. d. R. im Verzeichnis `/var/mail/` in einer Datei, die den Namen der Kennung trägt, mit der Sie sich angemeldet haben. Dies ist heutzutage aber die Ausnahme; daher laufen auf den Rechnern, die E-Mails empfangen, i. d. R. weitere Prozesse wie z. B. ein `POP3`- oder ein `IMAP`-Server, die einen Abruf der E-Mails über das Netzwerk ermöglichen. Wenn Ihr Mailprogramm nicht auf dem gleichen Rechner läuft, auf dem eingehende E-Mail gespeichert wird, nutzt es eines der beiden Protokolle POP3 oder IMAP, um eingegangene E-Mails abzurufen.

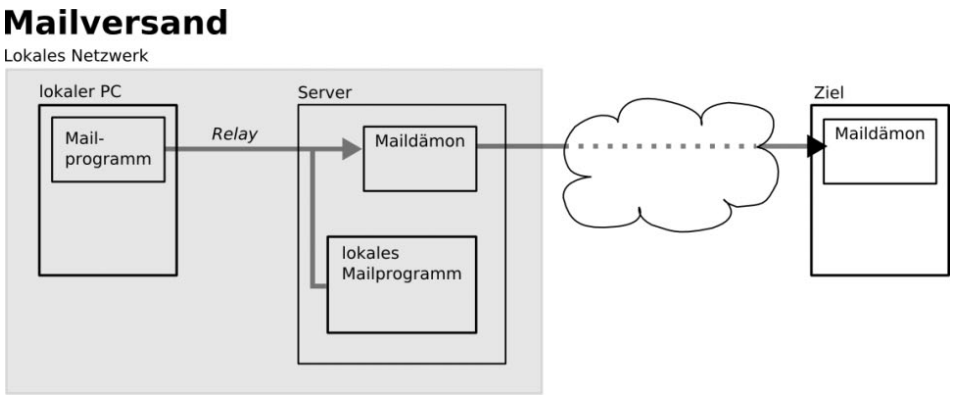


Abbildung 8.1: Mailversand

8.1.1 Grundlagen: Mailversand

Wenn Sie nicht mit dem laufenden Mailserver-Prozess direkt kommunizieren[2], arbeitet dieser *immer* als **Relay**, d. h. er leitet eine E-Mail weiter. Eine selten gewordene Ausnahme ist die Nutzung des Mailserver-Programms als Anwendung, die bei einem Aufruf die E-Mail direkt abliefert. Dies wird aber i. d. R. nur noch von wenigen Anwendungen auf Unix- und Linux-Systemen genutzt.

Die für den Server wichtige Frage ist jetzt, *von wem* er E-Mails annehmen darf und was er in welchem Fall damit tun muss:

- Der einfachste Fall: Die Anfrage kommt über die lokale Loopback-Schnittstelle. Das bedeutet i. d. R., dass das Programm auf dem gleichen Rechner läuft wie der Server selbst. Eine Weiterleitung ist in diesem Fall ungefährlich.
- Der zweiteinfachste Fall: Es kommt eine Anfrage von außen mit einer *lokalen* Adresse. Diese Mail soll der Server annehmen und speichern. Er muss nur wissen, wann die Adresse eine *lokale* Adresse ist: dies ist insbesondere dann wichtig, wenn der Server mit mehr als einem Namen angesprochen werden kann: diese Namen müssen dem Mailserver in der Konfiguration mitgeteilt werden.
- Kompliziert wird es, wenn der Mailserver auch E-Mails verarbeiten soll, die ihm von anderen Rechnern im Netzwerk übergeben werden, aber zu keiner lokalen Adresse gehören. Diese Funktion ist heutzutage in der Standardkonfiguration abgeschaltet. Wenn Sie also z. B. den Clients in Ihrem Netzwerk das direkte Versenden von E-Mails nicht erlauben wollen, können Sie den Mailserver so konfigurieren, dass er E-Mails von Rechnern aus dem lokalen Netzwerk nach draußen weiterleitet.

[2] Lachen Sie nicht, das funktioniert! Es existieren noch Gurus auf dieser Welt, die mit Hilfe von telnet eine Verbindung zu einem Mailserver herstellen, mit einem Mix aus Kommandos und Text E-Mails verfassen und diese dann versenden können.

8.1.2 Das Kreuz mit dem Spam

Am Anfang war die E-Mail, und alles war gut. Dann kamen findige Leute auf die Idee, dass man per E-Mail sehr kostengünstig Werbung verschicken könnte. Da es kein zentrales Verzeichnis für E-Mail-Adressen gab, musste man sich einen anderen Weg suchen, um an potentielle Empfänger heranzukommen. Dies war in den Anfangszeiten der Internet-Nutzung noch relativ einfach: So wurden z. B. in News-Beiträgen, die öffentlich zugänglich waren, die E-Mail-Adressen der Ersteller gespeichert. Der Werbetreibende musste also nur automatisiert alle News-Beiträge auf eine E-Mail-Adresse durchforsten und hatte eine schöne Grundlage für eine Rundmail zu einschlägigen Themen. Später wurden auch Webseiten einbezogen und andere Medien, die mit Hilfe von Suchmaschinen gefunden werden konnten.

Was macht der Spam-Versender nun mit den E-Mail-Adressen? Anfänglich konnte er sich einfach einen Rechner nehmen und von dort einen Brief an alle E-Mail-Adressen schicken. Das stieß natürlich den Empfängern sauer auf, und sie überlegten sich etwas dagegen: sehr schnell entstanden »schwarze Listen« mit IP-Adressen von Servern, die Spam versendeten. Aber ein richtiger Spammer lässt sich davon natürlich nicht beeindrucken: Die nächste Schlacht wurde mit Hilfe von Mailservern geschlagen, die auf Grund einer schlecht konfigurierten Relay-Funktion für die Verteilung missbraucht werden konnten. Und die armen, missbrauchten Server landeten prompt in den schwarzen Listen. Abhilfe schafften Erweiterungen, mit denen das Relaying eingeschränkt werden konnte. Interessanterweise wurde eine Funktion, die das Problem von Anfang an gelöst hätte, nur zögerlich eingesetzt – übrigens auch heute noch: die Autorisierung durch Kennung und Passwort *auch für das Versenden* von Mails. In der Regel ist es nämlich kein Problem, auf die für das Abholen der Mails vorgehaltenen Kennungen und Passwörter auch für das Versenden zurückzugreifen.

Für die Empfängerseite wurden mit Hilfe sogenannter *RBL*-Server (**R**ealtime **B**lackhole **L**ist) Funktionen realisiert, die dem Mailserver eine Entscheidungshilfe gaben, eine E-Mail anzunehmen oder abzulehnen. Vor der Annahme überprüft der Mailserver, ob die IP-Adressen des letzten Relays in der schwarzen Liste als Spam-Versender bekannt ist. Meldet der RBL-Server einen Spammer, kann der Empfänger-Server die Mails entweder ablehnen oder ohne besonderen Hinweis nach `/dev/null` entsorgen. Da Spam-E-Mails oft einen falschen Absender haben, wird i. d. R. Letzteres gewählt, um durch die zurückgehenden E-Mails das Netz nicht unnötig zu belasten.

Die RBL-Anbieter tun jedoch mehr, als nur Listen von bekannten Spammern zu führen: Zur Spam-Prävention testen sie erreichbare oder bekannte Server auf Lücken in der Mail-Konfiguration. So kann man, nur weil man seinen Mailserver falsch konfiguriert hat, trotzdem auf einer schwarzen Liste landen – ohne als Spammer auffällig geworden zu sein. Von so einer Liste herunterzukommen, ist deutlich schwieriger und vor allem langwieriger, als hinaufzukommen. In der Zwischenzeit kann von diesem Rechner aus keine E-Mail verschickt werden. Im günstigsten Fall merkt man, dass eigene Mails nicht mehr erwünscht sind; im ungünstigsten Fall bekommt man noch nicht einmal mit, dass eigene Mails den Empfänger nicht mehr erreichen.

Weniger Spam gibt es dadurch aber nicht. Sie werden sicher schon von den sogenannten Bot-Netzen gehört haben: Da die schwarzen Listen auf IP-Adressen basieren, musste der Spammer, der etwas auf sich hielt, feste IP-Adressen zugunsten variabler meiden. Von trojanischen Pferden oder Viren befallene Computer befinden sich i. d. R. in einem lokalen Netzwerk, in dem ihnen erlaubt ist, E-Mails über den lokalen Mailserver zu versenden. Werden solche Rechner von Spammern missbraucht, besteht die Gefahr, dass der lokale Mailserver zum Spam-Server wird, obwohl keine Fehlkonfiguration vorliegt.

Die schwarzen Listen haben an Bedeutung verloren durch von Viren und/oder Trojanischen Pferden befallene Rechner, die zu Bot-Netzen zusammengefasst für den E-Mail-Versand genutzt werden. In großen Netzwerken organisiert, werden auf viele verschiedene IP-Adressen verteilt Spam-Mails verschickt und machen damit schwarze Listen nutzlos. Um Spam effektiv abwehren zu können, muss der Mailserver jetzt auf der Basis der E-Mail selbst entscheiden, ob diese Spam darstellt oder nicht. Die zurzeit erfolgreichste Art der Bekämpfung setzt wiederum bei den Mailservern während des Empfangs der Mail an: Ähnlich einem Virenscanner wird eine Mail durch einen Spam-Erkenner geschickt, der Spam-Signaturen regelmäßig auffrischt und so i. d. R. in der Lage ist, bei großen Installationen die Spam-Last für die Anwender deutlich zu reduzieren.

8.1.3 Grundlagen: Mailempfang

Der Mail-Empfang könnte so geruhsam ablaufen: Es kommt eine Mail an, man schaut nach, an wen sie geht, und dann wird sie ins passende Postkörbchen gelegt. Mailserver können mit einer E-Mail aber noch eine ganze Menge mehr tun, bevor sie dazu übergehen, sie zu speichern:

- Die E-Mail selbst wird auf Spam getestet. Je nach System werden der E-Mail Header-Zeilen hinzugefügt, in denen ein Spam-Level angegeben wird. Dann kann der Anwender beim Abruf seiner Post nach den Headern filtern und unerwünschte E-Mails löschen oder in einen separaten Ordner verschieben. Dies hat aber den Nachteil, dass alle E-Mails heruntergeladen werden müssen, bevor eine Entscheidung pro oder contra Spam getroffen werden kann.
- Anhänge werden extrahiert und durch einen Virenscanner geschickt. Wird ein Virus gefunden, wird der Anhang von der Mail entfernt und zur Sicherheit in einem speziellen Bereich zwischengespeichert, sodass der Nutzer ihn noch abrufen kann, wenn er ihn unbedingt braucht.
- Eine andere Variante ist die Speicherung von als Spam klassifizierten Mails auf dem Server und die Versendung einer regelmäßigen Übersicht der so aussortierten elektronischen Post (ein sogenannter Digest). Dadurch lädt der Anwender nur eine E-Mail herunter, muss diese aber auf falsch positiv gekennzeichnete E-Mails durchsehen und hat die Mühe, die falsch gekennzeichnete Mail vom Server herunterladen zu müssen. Da die Spam-Mails aus Kapazitätsgründen nur eine begrenzte Zeit im Zwischenspeicher gehalten werden können, verschwinden falsch positiv gekennzeichnete Mails, wenn man sich nicht früh genug darum kümmert.

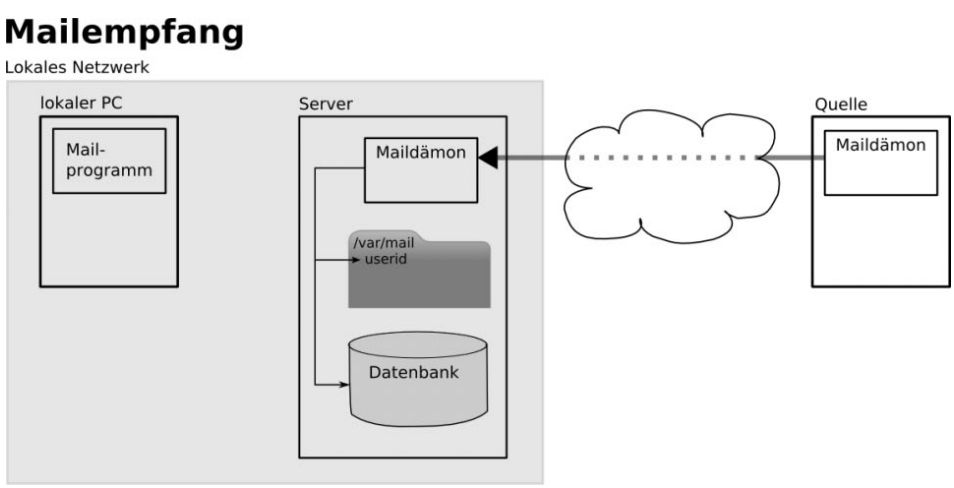


Abbildung 8.2: Mailempfang

Am Ende haben wir eine E-Mail, von der wir ausgehen müssen, dass der Empfänger sie gerne lesen möchte. Je nach Anzahl der verwalteten E-Mail-Nutzer und je nach dem für den Abruf genutzten Protokoll muss man sich nun Gedanken über die Art der Speicherung machen:

- Die Standardvariante sind einfache Mailbox-Dateien, die im Verzeichnis `/var/mail` vorgehalten werden, bis der Nutzer die darin enthaltenen E-Mails abruft.
- Neigen die Nutzer aber dazu, lange E-Mails zu bekommen oder diese lange in der Mailbox aufzubewahren, anstatt sie herunterzuladen, muss man sich einerseits überlegen, ob man das Dateisystem mit Hilfe von Mengenbeschränkungen vor einem Überlaufen schützt oder ob man die Mails stattdessen von einem Mailsystem speichern lässt, das neben einstellbaren Mailbox-Größen auch einen schnelleren Zugriff auf große Mailboxen bietet.
- Werden POP3- oder IMAP-Server für den Abruf der E-Mails eingesetzt, sollte man außerdem den Unterschied zwischen beiden Protokollen berücksichtigen: IMAP ist die modernere Variante, die auch das Anlegen von Ordnern und Unterordnern *auf dem Mailserver* ermöglicht – im Gegensatz zu POP3, das nur den Abruf erlaubt. Mit IMAP kann eine Mail-Ordner-Hierarchie verwaltet und von verschiedenen Clients aus genutzt werden. Dies bietet dem Nutzer Unabhängigkeit von einem, speziellen Arbeitsplatz-PC, stellt aber den Server-Administrator vor das Problem, für ausreichend Speicherplatz auf dem Server sorgen zu müssen.
- IMAP-Server existieren in verschiedenen Varianten: Einige verwenden das Home-Verzeichnis des Nutzers für die Speicherung der von ihm angelegten Mail-Ordner und Unterordner. Andere nutzen dafür ein eigenes Verzeichnis oder ein externes SQL-Datenbanksystem.
- Bei Systemen, die E-Mails in Datenbanken speichern, funktioniert aber i. d. R. der normale Weg der Mail-Annahme nicht mehr. Also muss für diese der Mailser-

ver so konfiguriert werden, dass die ankommenden E-Mails nicht in einer Datei in `/var/mail` landen, sondern in der jeweiligen Datenbank, inklusive der Überprüfung, ob dem Nutzer überhaupt noch genug Speicherplatz für die Annahme dieser E-Mail zur Verfügung steht. Mittlerweile bieten aber alle Mailserver Funktionen, um mit datenbankgestützten POP- oder IMAP-Servern zusammenarbeiten zu können.

8.1.4 Grundlagen: Mailabruf

Wie kommt der Empfänger nun an seine E-Mail heran? Er muss sie von dem Server abrufen, der sie für ihn gespeichert hat. Dazu müssen sowohl server- als auch clientseitig einige Voraussetzungen erfüllt sein.

- Auf dem Server, auf dem die E-Mails gespeichert sind, muss ein POP- oder ein IMAP-Server laufen.
- Dieser muss auf Kennungen und Passwörter zugreifen können und diese auch den jeweils gespeicherten E-Mails zuordnen können. Je nach System werden dazu die Unix-Systemkennungen genutzt oder aber völlig eigenständige Nutzerverwaltungen.
- Der Zugriff sollte gesichert durch SSL oder andere Verfahren erfolgen, insbesondere wenn der Zugang auch von außerhalb des lokalen Netzwerks möglich sein soll.
- Der Anwender benötigt die Zugangsdaten zu seinen E-Mails: in seinem Mailclient muss er den Server inklusive seiner Kennung und dem dazugehörigen Passwort eintragen. Dann muss er die Art des Zugriffs wählen; mittlerweile

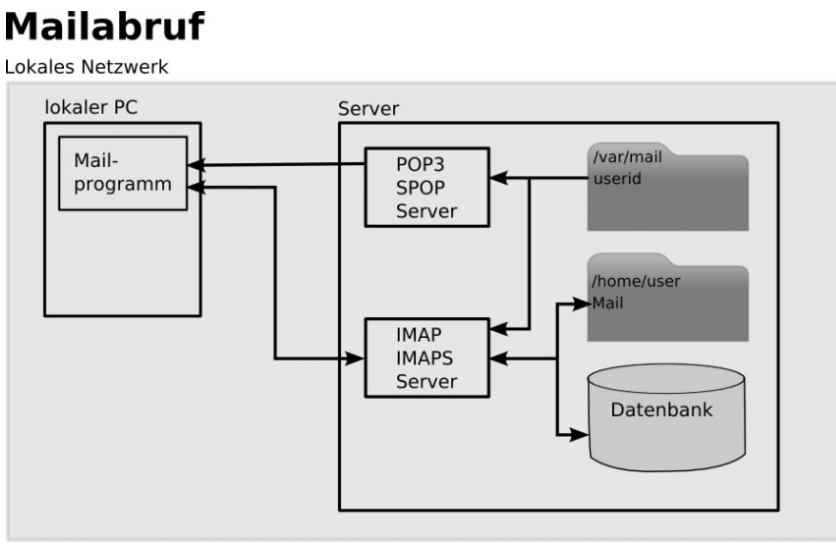


Abbildung 8.3: Mailabruf

bieten die meisten Mailclients eine Möglichkeit, »die Fähigkeiten des Servers zu testen« und so automatisiert die sicherste Alternative für den Download der Mails zu ermitteln.

- Je nach Spam- und Virenverarbeitung muss der Anwender die eingehende Mail dann nach bestimmten Regeln filtern, um einen Spam- und virenfreien Posteingang zu erhalten.
- Außerdem kann der Anwender das Mailprogramm so einstellen, dass es regelmäßig nachschaut, ob neue E-Mail angekommen ist. Die Frequenz, in der nachgeschaut werden darf, kann dabei je nach Provider bis zu 15 Minuten betragen.
- Bei der Nutzung von IMAP können auf dem Server Ordner angelegt und eingegangene E-Mails dauerhaft gespeichert werden.
- POP in seinen verschiedenen Varianten erlaubt nur das Herunterladen eingegangener Mails. Moderne Mailprogramme ermöglichen es zwar, die E-Mails auf dem Server zu belassen und schaffen es trotzdem, Buch über neue und bereits gelesene Mails zu führen. Ein aus diesem Grund überfüllter Posteingang auf dem Mailserver kann aber dazu führen, dass keine neuen E-Mails mehr angenommen werden und die Versender mit der für den Empfänger unangenehmen Meldung beglückt werden, die Mailbox sei voll. Den Einsatz dieser »Lösung der serverseitigen Speicherung von E-Mails« halten wir daher nur im Notfall für sinnvoll.

Das waren im Groben die Grundlagen, die für das Verständnis des Umgangs mit E-Mails aus der Sicht eines Server-Administrators wichtig sind. Widmen wir uns als Nächstes den wichtigsten Mailprogrammen für Linux.

8.2 Mailprogramme für Linux

E-Mail-Clients für Linux gibt es eine ganze Reihe. Wir wollen auf eine ausführliche Einführung in einen der Clients verzichten und uns auf einen Überblick beschränken, wie man mit welcher Software E-Mails verarbeiten kann.

X-basierte E-Mail-Clients

Den größten Komfort bei der Verarbeitung von E-Mails bieten sicher die Software-Suiten, die zu den großen Desktops gehören. Es gibt aber auch Nischenlösungen, die je nach Anwendungsbereich durchaus sinnvoll eingesetzt werden können. Die für den Abruf und den Versand von E-Mails notwendigen Protokolle wie POP3, SPOP, IMAP, IMAPS, SMTP und Authenticated SMTP verstehen alle der hier genannten Programme. Die meisten besitzen auch die Infrastruktur, um verschlüsselte E-Mails zu versenden und zu empfangen.

- Zu den integrierten Mail-Suiten gehören die Clients der großen Desktops: KMail bzw. Kontact kommen aus dem KDE-Umfeld, Evolution von den GNOME-Entwicklern. Beide stellen eine Outlook-ähnliche Umgebung mit E-Mail-Client, PIM-Funktionen und zusätzlichen Erweiterungen zur Verfügung.

- Den E-Mail-Client der Mozilla-Suite Thunderbird kann man natürlich auch unter Linux nutzen. Anwender, die mit diesem bereits unter Windows gearbeitet haben, werden zur Linux-Variante keinen Unterschied feststellen.
- Mit Sylpheed existiert ein leichtgewichtiger E-Mail-Client, der sich in seinen Funktionen auf das Empfangen und Versenden von Mails beschränkt und durch geringe Systemanforderungen und hohe Verarbeitungsgeschwindigkeit besticht.

Textbasierte E-Mail-Clients

Braucht man noch textbasierte E-Mail-Programme? Dies ist keine rein rhetorische Frage. Die Einsatzgebiete sind vielfältig, und je nach Client der Umgang durchaus komfortabel. Auf wirklich alten Rechnern oder für Anwender, die sowieso keinen Nutzen von graphischen Oberflächen haben wie z. B. Blinde oder Sehbehinderte kann der Einsatz eines textbasierten E-Mail-Clients durchaus lohnenswert sein.

- Der Star unter den textbasierten E-Mail-Clients ist Mutt. Er kann alles, was auch die »großen« E-Mail-Clients bieten, bis zu `IMAP` und der Einbindung von PGP/GPG für die Verschlüsselung von E-Mails.
- Ein weiterer, mächtiger E-Mail-Client ist Pine. Dieser Client ist aus lizenzrechtlichen Gründen nicht in Debian oder Fedora integriert, es lassen sich aber passende Pakete über die Homepage `http://www.washington.edu/pine/` herunterladen.
- Wer sowieso statt des Linux-Kernels (oder mit dem Linux-Kernel) bereits den Editor Emacs gebootet hat[3], kann mit ihm auch direkt E-Mail lesen.
- Auf der Kommandozeile kann man auch mit den Kommandos `mail` bzw. `mailx` die lokale Mailbox bearbeiten oder E-Mails verschicken. Wir werden diese Programme weiter unten zu Demonstrationszwecken verwenden, da sie i. d. R. zu einer Basisinstallation gehören und auf jedem Linux-System zu finden sein sollten. Mit wenigen Eingaben kann man damit eine E-Mail versenden:

```
victor@hugo:~$ mail -s 'Ein Subject' victor
Huhu!
 STRG + D
CC: Return
victor@hugo:~$
```

- Aus der BSD-Welt hat sich auch das Programm from in die Linux-Welt gerettet. Es zeigt bei einem Aufruf einfach die Absender der im Eingangspostfach liegenden E-Mails an.

Webbasierte E-Mail-Clients Auf Services, wie sie GMX oder Web.de bieten, muss man auch unter Linux nicht verzichten: Mit Horde/IMP steht ein mächtiges Webmail-System zur Verfügung, das sogar in der Lage ist, `IMAP`-Ordner zu verwalten.

E-Mails mit Programmiersprachen verarbeiten Python und Perl bieten Bibliotheken, die die Verarbeitungsprozesse rund um das Mailen unterstützen. Beispiel Python:

[3] Kleiner Scherz. Aber selbst Richard Stallman ist sich nicht mehr sicher, ob es sich bei seinem Emacs noch um eine Software oder schon um ein Betriebssystem handelt.

- Die Bibliothek `smtplib` erlaubt das Verschicken von E-Mails.
- Die Bibliothek `email` bietet Funktionen, um Mails zu verarbeiten, Anhänge zu extrahieren und neue Mails zusammenzustellen. Sie löst die aus den älteren Python-Versionen bekannten Bibliotheken `rfc822`, `mimetools` und `multifile` ab.
- Des Weiteren stehen Bibliotheken zur Verfügung, die Anhänge codieren und decodieren können: `base64`, `binascii`, `binhex`, `quopri` und `uu`.
- Mit `mailbox` und `mhlib` können lokale Mailboxen verschiedenen Typs gelesen werden.
- Und schließlich bieten `mailcap` und `mimetypes` die Möglichkeit, auf Anhänge mit spezifizierten Mime-Typen passend zu reagieren.

Und natürlich: Fetchmail! Bei Einwahlverbindungen oder Mailboxen, die per POP oder IMAP irgendwo im Netzwerk vorgehalten werden, sorgt Fetchmail für das Herunterladen der E-Mails und die lokale Verteilung. Die Software lädt vom externen Provider die E-Mail herunter und gibt sie an den lokalen Mailserver weiter. Bietet der Provider die Möglichkeit, verschiedene E-Mail-Adressen in einer Mailbox zu speichern, bietet Fetchmail die Möglichkeit, diese auf verschiedene lokale Postfächer zu verteilen. Als Nicht-GUI-Programm lässt sich Fetchmail gut in automatische Einwahlprozeduren einbinden.

8.3 E-Mails intern: das Format elektronischer Post

Bevor wir uns der Einrichtung eines Mailservers widmen, wollen wir noch einen Blick auf den Aufbau einer E-Mail werfen: Der Austausch von Informationen über das SMTP-Protokoll geschieht i. d. R. auf der Basis von ASCII-Zeichen im 7-Bit-Modus. Es existiert zwar mittlerweile ein 8-Bit-Modus, jedoch werden auch binäre Daten vor der Übermittlung in »lesbare« Zeichen umgesetzt, sodass der Quelltext einer E-Mail mit einem Texteditor gelesen und – bis auf die Attachments – auch verstanden werden kann.

Eine E-Mail besteht aus zwei Teilen: einem *Header* und einem *Body*. Im Header werden z. B. der Absender, der Empfänger und der Betreff gespeichert. Der Header endet mit einer Leerzeile. Danach folgt der Body, der entweder aus dem Text der E-Mail oder aus den Teilen einer kombinierten E-Mail mit Attachment besteht. Im folgenden Beispiel schicken wir an die Kennung, mit der wir uns angemeldet haben, eine kurze Mail und schauen uns dann das Ergebnis in der Eingangsmailbox an:

```
victor@hugo:~$ mail -s 'Betreff' victor@hugo
Body
Cc:
victor@hugo:~$ cat /var/mail/victor
From victor@hugo Sun Apr 30 16:57:20 2006
Return-path: <victor@hugo>
Envelope-to: victor@localhost
```

```
Delivery-date: Sun, 30 Apr 2006 16:57:20 +0200
Received: from victor by localhost with local (Exim 4.50)
        id 1FaDMO-0001Id-FY
        for victor@localhost; Sun, 30 Apr 2006 16:57:20 +0200
To: victor@localhost
Subject: Betreff
Message-Id: <E1FaDMO-0001Id-FY@localhost>
From: Victor Y. Secosbosque <victor@hugo>
Date: Sun, 30 Apr 2006 16:57:20 +0200

Body
victor@hugo:~$
```

Ein paar kurze Hinweise zu den wichtigsten Punkten, die man beim Lesen des Quellcodes einer Mail beachten muss:

- Wenn die E-Mail in einer Mailbox abgelegt ist, wurde ihr eine Zeile vorangestellt, die mit einem `From` *ohne* folgenden Doppelpunkt beginnt. Damit es keine Verwirrung gibt, werden andere Zeilen, die zufällig mit `From` anfangen, durch ein >-Zeichen am Zeilenanfang »entwertet«.
- Jede Header-Zeile beginnt mit einem Schlüsselwort, gefolgt von einem Doppelpunkt und dem dazugehörigen Text. Geht der Text über mehrere Zeilen, muss er eingerückt werden. Nur Schlüsselwörter dürfen am Anfang einer Zeile stehen – außer natürlich im Body.
- Die meisten Header-Einträge erklären sich von selbst. Auf die `Message-Id` sollten wir aber noch hinweisen: Diese wird vom versendenden Mailserver vergeben und ist (weltweit und für alle Zeiten, wenn alles gut geht) eindeutig. Sie wird von den Mailclients genutzt, um Verbindungen zwischen E-Mails herzustellen.

Dies war ein vergleichsweise einfaches Beispiel. Um Ihnen einen Eindruck zu vermitteln, wie eine »normale« E-Mail aussehen kann, die über das Netz von Server zu Server gereicht wurde, zeigen wir Ihnen hier noch eine E-Mail, die über eine Mailingliste an uns versendet wurde. Mit den oben genannten Informationen sollten Sie zumindest in der Lage sein, die einzelnen Elemente der E-Mail zu erkennen:

```
From bounce-debian-user-german=victor=provider@lists.debian.org ttt mmm nn 2006
Return-path: <bounce-debian-user-german=victor=provider@lists.debian.org>
Envelope-to: victor@localhost
Delivery-date: ttt, nn mmm 2006 21:34:13 +0200
Received: from localhost ([127.0.0.1] helo=hugo)
        by hugo with esmtp (Exim 4.61)
        (envelope-from <bounce-debian-user-german=victor=provider@lists.debian.org>)
        id 1FZvCm-0006HB-Tj
        for victor@localhost; ttt, nn mmm 2006 21:34:12 +0200
X-Flags: 0000
Delivered-To: Provider delivery to victor@provider
```

```
Received: from pop.provider [nn.nn.nn.nn]
        by hugo with POP3 (fetchmail-6.3.2)
        for <victor@localhost> (single-drop); ttt, nn mmm 2006 21:34:12 +0200 (CEST)
Received: (qmail invoked by alias); ttt, nn mmm 2006 19:30:50 -0000
Received: from murphy.debian.org (EHLO murphy.debian.org) [nn.nn.nn.nn]
  by provider with SMTP; ttt, nn mmm 2006 21:30:50 +0200
Received: from localhost (localhost [127.0.0.1])
        by murphy.debian.org (Postfix) with QMQP
        id 164B92DF25; ttt, nn mmm 2006 14:30:36 -0500 (CDT)
Old-Return-Path: <ab@sender>
X-Original-To: debian-user-german@lists.debian.org
Received: from irgendwo (irgendwo [nn.nn.nn.nn])
        by murphy.debian.org (Postfix) with ESMTP id 130772DD8F
        for <debian-user-german@lists.debian.org>; ttt, nn mmm 2006 14:30:29 -0500 (CDT)
Received: from dsl-anbieter ([nn.nn.nn.nn] helo=[nn.nn.nn.nn])
        by irgendwo with esmtpa (Exim 4.50)
        id 1FZv9A-0007gi-8p
        for debian-user-german@lists.debian.org; ttt, nn mmm 2006 21:30:29 +0200
Message-ID: <4453BECA.2030002@sender>
Date: ttt, nn mmm 2006 21:30:18 +0200
From: =?ISO-8859-15?Q?A.B._Sender?= <ab@sender>
User-Agent: Mail/News 1.5 (X11/20060228)
MIME-Version: 1.0
To: german debian users <debian-user-german@lists.debian.org>
Subject: Re: Ein Problem
References: <BAY137-7947C3B30F278409A7834F7DE0E1@anderswo>
In-Reply-To: <BAY137-7947C3B30F278409A7834F7DE0E1@anderswo>
X-Enigmail-Version: 0.94.0.0
OpenPGP: id=E1N31D
Content-Type: multipart/signed; micalg=pgp-sha1;
 protocol="application/pgp-signature";
 boundary="------------hubaBAC16296916C865C6F010F8A"
X-Rc-Virus: 2007n-nn-nn_01
X-Rc-Spam: 2007n-nn-nn_01
X-Spam-Checker-Version: SpamAssassin 3.0.3 (2005-04-27) on murphy.debian.org
X-Spam-Status: No, score=-4.8 required=4.0 tests=AWL,LDOSUBSCRIBER
        autolearn=no version=3.0.3
Resent-Message-ID: <gcNanB.A.DyD.c77UEB@murphy>
Resent-From: debian-user-german@lists.debian.org
X-Mailing-List: <debian-user-german@lists.debian.org> archive/latest/nnnnnn
X-Loop: debian-user-german@lists.debian.org
List-Id: <debian-user-german.lists.debian.org>
List-Post: <mailto:debian-user-german@lists.debian.org>
List-Help: <mailto:debian-user-german-request@lists.debian.org?subject=help>
List-Subscribe: <mailto:debian-user-german-request@lists.debian.org?subject=subscribe>
```

```
List-Unsubscribe: <mailto:debian-user-german-request@lists.debian.org?subject=unsubscribe>
Precedence: list
Resent-Sender: debian-user-german-request@lists.debian.org
Resent-Date: ttt, nn mmm 2006 14:30:36 -0500 (CDT)
X-???-Antivirus: -1 (not scanned, may not use virus scanner)
X-???-Antispam: -2 (not scanned, spam filter disabled)
X-???-UID: lrSViNATlIvOo7n6WVGU2poZZl09I4LF

This is an OpenPGP/MIME signed message (RFC 2440 and 3156)
--------------hubaBAC16296916C865C6F010F8A
Content-Type: text/plain; charset=ISO-8859-15
Content-Transfer-Encoding: quoted-printable

Jemand schrieb:
> Etwas

Text

--=20
Gruß,
A.B. Sender


--------------hubaBAC16296916C865C6F010F8A
Content-Type: application/pgp-signature; name="signature.asc"
Content-Description: OpenPGP digital signature
Content-Disposition: attachment; filename="signature.asc"

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.3 (GNU/Linux)
Comment: Using GnuPG with GroßesMailprogramm - http://...

iD8DBQFEU77KNOOTf/FLKYsRAq/BAJ9cMxsy1YIo3BipGxnb4aVkcFC6iACgvBsd
EA+tRXBvQaO8AZN8VZVRWXQ=
=CSVG
-----END PGP SIGNATURE-----

--------------hubaBAC16296916C865C6F010F8A--
```

Beginnen wir mit dem Teil des Headers, der als Erstes auffällt:

- den `Received`-Zeilen. Jeder Mailserver, der eine E-Mail annimmt, um sie entweder weiterzuschicken oder lokal zu speichern, verewigt sich mit (mindestens) einer `Received`-Zeile. Wir haben die Zeitstempel im Beispiel zwar anonymisiert, die Uhrzeit aber stehen gelassen, um Ihnen eine Möglichkeit zu geben, die Reihenfolge zu erkennen, in der die Received-Zeilen angelegt werden. Ignorieren

wir einmal die ersten vier Zeilen des Headers, die beim Speichern der E-Mail angelegt werden, stellen wir fest, dass jeder Mailserver seinen `Received`-Eintrag am *Anfang* der E-Mail anfügt.

- Wenn Sie die `Received`-Einträge genau ansehen, werden Sie feststellen, dass auch Unterprozesse auf dem gleichen Server sich mit einem Eintrag verewigen. So meldet sich der Mailinglisten-Server `murphy.debian.org` mit zwei Einträgen. Im unteren Bereich finden Sie dann eine Header-Zeile `X-Spam-Checker-Version`, die darauf hinweist, dass auf diesem Server ein SpamAssassin installiert ist, an den die E-Mail zur Überprüfung übergeben wurde.
- Überhaupt finden sich in dieser E-Mail eine ganze Reihe zusätzlicher Header-Zeilen. Hier können wir zwei Arten unterscheiden: Die Zeilen mit einem »X-« am Anfang sind von Software, die mit der E-Mail in Berührung gekommen ist, für »den Eigengebrauch« eingefügt worden. Im unteren Bereich finden Sie z. B. `X-???-`Zeilen, die wir ebenfalls anonymisiert haben, da sich in den drei Fragezeichen im Original der Provider verewigt hat. Weiter oben finden Sie `X-Spam-Status`, das vom SpamAssassin gesetzt wird und von einem Mailclient genutzt werden kann, um als Spam markierte E-Mails auszusortieren.
- Da es sich um eine E-Mail handelt, die über eine Mailingliste gegangen ist, sind auch Spuren der Mailinglisten-Software zu entdecken: die Header mit `List-` enthalten einige Zusatzinformationen zur Mailingliste. Die Thread-Darstellung im Mailclient wird jedoch nicht über `List`-Einträge, sondern über die Header-Zeilen `References` und `In-Reply-To` abgewickelt.
- Zum Schluss noch ein Blick auf den Body: Sie können anhand des Header-Feldes `Content-Type` erkennen, dass es sich um eine mehrteilige (`multipart/mixed`) E-Mail handelt. Im Header ist ein Begrenzungsstring angegeben, den Sie im Body wiederfinden. Dieser dient dazu, die einzelnen Teile der E-Mail voneinander zu trennen. Wir haben es hier mit einer von OpenPGP signierten E-Mail zu tun; der erste Teil enthält den Text, der zweite Teil einen mit dem Schlüssel des Versenders erstellten Code, mit dem sich die Echtheit des Inhalts überprüfen lässt. Eine E-Mail könnte an jedem Punkt der Übergabekette abgefangen und verändert worden sein. Bei einer signierten E-Mail können Sie immerhin sicher sein, dass der Teil innerhalb des Containers dem Ausgangszustand entspricht.
- Datei-Anhänge werden auf die gleiche Art und Weise abgehandelt: Sie finden im Kopf der E-Mail den Content-Type `multipart/mime` und einen Trenner. Im Body werden jetzt der E-Mail-Text und alle Dateien aufgeführt, begrenzt durch den Trenner. Vor den Daten des jeweiligen Containers finden Sie dann Header-Zeilen mit Spezifikationen für den `Content-Type` und das `Content-Transfer-Encoding`, das beschreibt, auf welche Art binäre Dateien mailfähig kodiert worden sind.
- Verfassen Sie E-Mails im HTML-Format, sorgen vernünftig programmierte E-Mail-Clients dafür, dass neben der HTML- auch eine Textversion mitgeschickt wird. Solche E-Mails sind ebenfalls vom Typ `multipart/mixed` und enthalten einen Text- und einen HTML-Teil.

8.3.1 Formate von Mailboxen

Kurz noch ein Blick auf zwei der gängigsten Mailbox-Formate: Die Mailserver legen die eingegangenen E-Mails in der Standardkonfiguration im BSD-Mailbox-Format, kurz mbox-Format, an. Dieses Format lässt sich sehr einfach erkennen und ist insbesondere auch dem menschlichen Auge noch zugänglich, wenn man sich die Datei mit einem Texteditor oder Ähnlichem ansieht. Wir schicken uns deswegen eine kurze E-Mail und werfen dann noch einmal einen Blick in unsere Eingangsmailbox:

```
victor@hugo:~$ mail -s 'Noch ein Betreff' victor@localhost
Etwas mehr Text
Cc:
victor@hugo:~$ cat /var/mail/victor
From victor@hugo Ttt Mmm dd 16:57:20 2006
Return-path: <victor@hugo>
Envelope-to: victor@localhost
Delivery-date: Ttt, dd Mmm 2006 16:57:20 +0200
Received: from victor by localhost with local (Exim 4.50)
        id 1FaDMO-0001Id-FY
        for victor@localhost; Ttt, dd Mmm 2006 16:57:20 +0200
To: victor@localhost
Subject: Betreff
Message-Id: <E1FaDMO-0001Id-FY@localhost>
From: Victor Y. Secosbosque <victor@hugo>
Date: Ttt, dd Mmm 2006 16:57:20 +0200

Body

From victor@hugo Ttt Mmm dd 17:40:21 2006
Return-path: <victor@hugo>
Envelope-to: victor@localhost
Delivery-date: Ttt, dd Mmm 2006 17:40:21 +0000
Received: from victor by hugo with local (Exim 4.50)
        id 1FaWlJ-0001Ka-BC
        for victor@localhost; Ttt, dd Mmm 2006 17:40:21 +0000
To: victor@localhost
Subject: Noch ein Betreff
Message-Id: <E1FaWlJ-0001Ka-BC@hugo>
From: Victor Y. Secosbosque <victor@hugo>
Date: Ttt, dd Mmm 2006 17:40:21 +0000

Etwas mehr Text
victor@hugo:~$
```

Neue E-Mails werden einfach hinter die bereits in der Datei vorhandenen kopiert. Als Anfangsmarkierung für jede Mail fungiert die schon genannte `From`-Zeile. Je

nach Programm wird zur besseren Trennung vor der `From`-Zeile noch eine Leerzeile eingefügt. Die großen Mail-Suiten nutzen ebenfalls das mbox-Format, legen aber zusätzlich Indexdateien an, um bei großen mbox-Dateien den Zugriff auf einzelne E-Mails zu beschleunigen. Dies ist auch das Hauptproblem des mbox-Formats: Je größer die Datei und je mehr E-Mails darin gespeichert sind, desto langsamer wird der Zugriff auf eine einzelne E-Mail. Außerdem muss beim Löschen einzelner E-Mails innerhalb einer mbox-Datei eine Art Garbage-Collection laufen. Sie müssen diesen i. d. R. manuell anfordern, bei KMail z. B. über die Option ORDNER KOMPRIMIEREN im Kontextmenü eines Mailordners. Vorsicht: Wenn beim Reorganisieren einer mbox etwas schiefgeht, weil beispielsweise der Speicherplatz nicht mehr ausreicht, können Mails verloren gehen!

Um das Zugriffsproblem zu lösen und auch den konkurrierenden Zugriff mehrerer Programme auf die E-Mails zu ermöglichen, wurde das Maildir-Format entwickelt. Hier besteht eine Mailbox aus einem Unterverzeichnis, das drei weitere Unterverzeichnisse enthält, in denen die E-Mails dann unter einem aus Ziffern und Buchstaben generierten, eindeutigen Namen gespeichert sind:

```
victor@hugo:~$ find Maildir
Maildir/
Maildir/tmp
Maildir/new
Maildir/new/1146485026.H693434P5790.hugo
Maildir/cur
victor@hugo:~$ cat Maildir/new/1146485026.H693434P5790.hugo
Return-path: <victor@hugo>
Envelope-to: victor@localhost
Delivery-date: Ttt, dd Mmm 2006 16:57:20 +0200
Received: from victor by localhost with local (Exim 4.50)
        id 1FaDMO-0001Id-FY
        for victor@localhost; Ttt, dd Mmm 2006 16:57:20 +0200
To: victor@localhost
Subject: Betreff
Message-Id: <E1FaDMO-0001Id-FY@localhost>
From: Victor Y. Secosbosque <victor@hugo>
Date: Ttt, dd Mmm 2006 16:57:20 +0200

Body
victor@hugo:~$
```

Im Beispiel sehen Sie die drei Unterverzeichnisse, die das Mailbox-Verzeichnis `Maildir` enthält: `cur` enthält die eingegangenen und bereits gelesenen E-Mails. Wenn Sie mit KMail das Maildir-Format nutzen, werden Sie die E-Mails als einzelne Dateien im Verzeichnis `cur` finden. Im Beispiel haben wir eine Funktion des Mailservers Exim4 genutzt, eingehende E-Mails in einem `Maildir`-Ordner im Homeverzeichnis der Kennung zu speichern. Diese werden von Exim4 dann im Verzeichnis `new` ge-

speichert. Die Mailfilter procmail und maildrop können ebenfalls mit dem Maildir-Format umgehen. Versuchen Sie jedoch nicht, eine E-Mail mit Hilfe dieser Filter direkt in eine von KMail angelegte Hierarchie zu kopieren. Auf Grund der internen Verarbeitung findet KMail die nicht von ihm selbst in den Mailboxen gespeicherten E-Mails nicht!

Es existieren noch weitere Formate für Mailboxen, von denen vielleicht noch das MH-Format, das ähnlich wie Maildir mit Unterverzeichnissen arbeitet, eine gewisse Bedeutung erlangt hat. Mittlerweile existieren jedoch für die meisten Mail- und vor allem IMAP-Server Anbindungsmöglichkeiten an Datenbanken. So können dann auch große Mengen an E-Mail performant verwaltet werden.

8.4 Installation eines Mailservers

Jedes Linux-System benötigt für interne Zwecke einen Mailserver. Viele als Dämonen laufende Prozesse benachrichtigen den Administrator über Besonderheiten mit Hilfe von E-Mails. Daher wird in jeder Distribution bei der Basis-Installation ein Mailserver mit installiert. Wir werden im Folgenden den bei Debian genutzten Mailserver Exim4 beschreiben. Auch wenn sich die von anderen Distributionen genutzten Mailserver in der Konfiguration zum Teil deutlich unterscheiden, haben doch alle Sysadmins mit den gleichen Themen zu kämpfen.

Die Paket-Übersicht zeigt, welche Pakete Sie für eine Exim4-Installation benötigen:

Installation eines Mailservers	Debian	Fedora	SUSE
Grundfunktionen Einfacher Mailserver, hauptsächlich für den internen Mailverkehr, der von den System-Tools benötigt wird	exim4-daemon-light	—	—
Vollwertiger Mailserver Exim in der »Vollversion«: mit zusätzlich einkompilierten Funktionen wie der Nutzung von Datenbanken für Mail-Aliase etc., die für einen mittleren bis großen Mailserver benötigt werden	exim4-daemon-heavvy	exim	exim
Monitor f. den Mailserver Monitoring-Tool mit grafischer Oberfläche	eximon4	exim-mon	eximon
mail(x) Einfache, kommandozeilenorientierte Mailclients	mailx	mailx	mailx

Tabelle 8.1: Paketübersicht

Vor das Installieren hat der Paket-Betreuer aber noch die Entscheidung darüber gesetzt, welche Funktion der gerade installierte Mailserver ausüben soll. Je nach Distribution erleichtern die Installationstools mit Hilfe von Szenarien die Konfiguration. Für diesen Moment reicht es aus, den Mailserver so zu konfigurieren, dass E-Mail nur lokal ausgeliefert wird.

Achtung

Wir demonstrieren Ihnen die Möglichkeiten anhand der Debian-Mechanismen. Leider besitzen weder Fedora noch SUSE Konfigurationstools für Exim4, so dass Sie dort die Konfigurationsdatei von Hand ändern oder bei den vorinstallierten MTAs bleiben müssen.

Bei der Erstinstallation werden Ihnen die folgenden Fragen automatisch gestellt; wollen Sie Ihren Mailserver später anders konfigurieren, können Sie dies über den Aufruf `dpkg-reconfigure exim4-config` tun, so wie wir es hier im Beispiel zeigen[4]:

```
hugo:~# dpkg-reconfigure exim4-config
Konfiguriere Exim v4 (exim4-config)
-----------------------------------

Die Debian-Exim4-Pakete können entweder eine große
(/etc/exim4/exim4.conf.template) oder rund 40 kleine Dateien in
/etc/exim4/conf.d/ als Datenquelle zum Erstellen der endgültigen Konfiguration
verwenden.

Ersteres eignet sich besser für größere Modifikationen und ist grundsätzlich
robuster, zweiteres ermöglicht es, mit geringem Aufwand kleine Änderungen
vorzunehmen, ist allerdings auch anfälliger und könnte bei weit gehenden
Veränderungen nicht mehr funktionieren.

Im Zweifelsfall sollten Sie sich gegen die Aufteilung in kleine Dateien
entscheiden.

Konfiguration auf kleine Dateien aufteilen? n


Wählen Sie die Konfigurationsart, die Ihren Bedürfnissen am besten entspricht.

Systeme mit wechselnder IP-Adresse (z.B. Dial-In-Zugang) sollten ausgehende
E-Mails zur Zustellung an einen Relay-Server (»Smarthost«) weitergeben. Sie
können wählen, ob ein derartiges System E-Mail empfangen und lokal zustellen
oder ob nur E-Mail für root und postmaster lokal zugestellt werden soll.

  1. Internet-Server - E-Mail wird direkt über SMTP versandt und empfangen
  2. Versand über Smarthost (Relay); Empfang mit SMTP oder fetchmail
```

[4] Wir haben das Frontend readline benutzt, da es sich im Buch einfacher darstellen lässt. Bei Ihrem Debian-System werden die Informationen mit größter Wahrscheinlichkeit über das Dialog-Frontend abgefragt, dessen Bedienung etwas einfacher ist.

```
  3. Versand über Relay-Host ohne lokale Zustellung normaler E-Mail
  4. Nur lokale E-Mailzustellung (keine Netzwerkverbindung)
  5. Manuelles Übernehmen einer bestehenden Exim v3 Konfiguration
  6. Keine Konfiguration zum jetzigen Zeitpunkt

E-Mail-Konfigurationsart: 4
```

Mit der zweiten Frage legen wir das Szenario fest, in dem wir uns bewegen wollen: Je nach Auswahl verändern sich danach die weiteren Fragen. Für den Anfang begnügen wir uns mit dem 4. Punkt, der lokalen Mailzustellung.

```
Der »E-Mailname« ist der dem Nutzernamen und '@'-Zeichen folgende Teil der
Adresse. Er taucht in ausgehender E-Mail auf, wenn er nicht mit rewriting
verborgen wird.

Dieser Name wird auch von anderen Programmen genutzt, es sollte der eindeutige
voll-qualifizierte Domainname (FQDN) dieses Rechners sein, er ist i.d.R. Teil
der Absender-Adresse lokal erzeugter E-Mails.

Dieser Name taucht nicht im Absender (From:) ausgehender E-Mail auf, wenn
»rewriting« aktiviert wird.

E-Mailname des Systems: hugo
```

Auf einem Server ohne Netzanschluss reicht es aus, hier den Namen des Servers anzugeben. Ansonsten ist die Angabe des vollqualifizierten Domainnamens Pflicht!

```
Geben Sie eine durch Doppelpunkte getrennte Liste von IP-Adressen an, auf denen
Exim lauschen soll. Sie müssen die Doppelpunkte innerhalb von IPv6-Adressen
verdoppeln (z.B. 5f03::1200::836f::::).

Wenn Sie keine Adressen angeben und die Liste leer lassen, wird Exim auf allen
verfügbaren Netzwerkschnittstellen eingehende SMTP-Verbindungen beantworten.

Wenn dieser Computer nicht direkt E-Mail per SMTP von anderen Rechnern empfängt,
sondern nur von lokalen Programmen wie fetchmail oder Ihrem Mailprogramm, die
sich mit »localhost« verbinden, sollten Sie das Verbinden externer Rechner zu
Exim unterbinden, indem Sie diese Option auf 127.0.0.1 setzen und dadurch
verhindern, dass Exim überhaupt auf den externen Netzwerkschnittstellen lauscht.

IP-Adressen, auf welchen Exim eingehende SMTP-Verbindungen beantwortet: 127.0.0.1
```

Wir geben hier nur die Adresse der Loopback-Schnittstelle an, da wir uns auf eine lokale Zustellung beschränken wollen. Wenn Ihr Server ans Netzwerk angeschlossen ist, Sie aber keine E-Mail nach draußen verschicken wollen oder sich bei der Konfiguration unsicher sind, können Sie hier ebenfalls die Loopback-Schnitt-

stelle angeben. Dann bleibt auch bei falsch konfiguriertem Exim4 alles »in der Familie«.

```
Bitte geben Sie eine Liste der Domänen an, für die dieser Rechner sich als
endgültiges Ziel betrachten soll, abgesehen vom lokalen Rechnernamen (Theodor)
und »localhost«.

Laut Voreinstellung werden alle angeführten Domänen gleich behandelt; wenn
unterschiedliche Domänen unterschiedlich behandelt werden sollen, müssen Sie die
Konfigurationsdateien bearbeiten.

Wenn ja, führen Sie sie hier in Form einer durch Doppelpunkte getrennten Liste
an. Andernfalls sollten Sie das Feld einfach leer lassen.

Weitere Domänen, für die E-Mail angenommen werden soll:
```

Weitere Domänen brauchen wir vorerst nicht anzugeben. Wenn Ihr Server jedoch unter mehreren Domainnamen erreichbar ist, müssen alle, für die er sich zuständig fühlen soll, hier eingetragen werden.

```
Normalerweise führt Exim diverse DNS Abfragen durch; beim Start, beim Empfangen
oder beim Zustellen von Nachrichten, für die Logdatei und um die Anzahl fest
eingetragener Werte in der Konfiguration klein zu halten.

Wenn dieser Rechner keinen dauerhaften Zugang zu DNS-Servern hat und sich bei
versuchtem Zugriff auf das Netz automatisch einwählt (Dial-on-Demand), kann
somit das Starten von Exim oder das Abarbeiten der Warteschlange (sogar, wenn
diese leer ist) zu einer kostenverursachenden Einwahl führen.

Aktivieren Sie die Option, wenn Sie automatische Einwahl (Dial-on-Demand)
verwenden, deaktivieren Sie sie andernfalls.

DNS-Anfragen minimieren (Automatisches Einwählen/Dial-on-Demand)? j
```

Diese Option schalten wir an, damit der Mailserver keine unnötigen DNS-Abfragen durchführt. Dies ist aber nur für einen lokalen Mailserver sinnvoll. Sobald der Mailserver E-Mails »von draußen« annimmt, sollte diese Funktion *ausgeschaltet* sein.

```
E-Mail für »postmaster«, »root« und andere Systemkonten wird normalerweise zum
Benutzerkonto des Systemadministrators weitergeleitet. Bleibt dieses Feld leer,
wird diese E-Mail in /var/mail/mail gespeichert, was allerdings nicht empfohlen
wird. Zumindest die E-Mails für postmaster sollten lokal gelesen und nicht auf
ein anderes System weitergeleitet werden, daher sollte zumindest einer der
Benutzer, die Sie angeben, seine E-Mails nicht weiterleiten. Verwenden Sie den
Präfix »real-« um lokale Zustellung zu erzwingen.
```

```
Führen Sie einen oder mehrere Benutzernamen getrennt durch Leerzeichen an.

Empfänger der E-Mails für root und postmaster: victor
```

Es hat sich eingebürgert, auf jedem Computer, der einen Mailserver beherbergt, die Adresse `postmaster` einzurichten. An diese werden E-Mails adressiert, die auf Probleme mit dem Mail-Versand hinweisen, sei es nun von intern laufenden Prozessen (eher selten) oder von externen Nutzern bzw. System-Administratoren. Damit Mails, die an diese Adresse geschickt werden, nicht »untergehen«, sollten diese an eine unprivilegierte Kennung desjenigen weitergeleitet werden, der diesen Rechner betreut. Dies gilt ebenfalls für Mails, die an die root-Kennung geschickt werden. Alle Konfigurationstools fragen Sie daher nach einer normalen Nutzerkennung. Wenn es sich um Ihren Arbeitsplatz-Rechner handelt, geben Sie dort die Kennung an, die Sie normalerweise nutzen. Sie können aber auch eine Mailadresse auf einem anderen Rechner angeben. Beachten Sie jedoch, dass bei Fehlfunktion des Mailsystems auch die Mails, in denen der Fehler gemeldet wird, nicht ankommen. :-)

```
Restarting MTA: exim4.
hugo:~#
```

Damit ist die einfache Grundkonfiguration für einen internen Mailserver abgeschlossen. Im Folgenden wollen wir mit ein paar Tests herausbekommen, ob er auch funktioniert.

8.5 Test einer Server-Installation

Sie können mit Hilfe des telnet-Programms einen einfachen Test durchführen, ob der Mailserver läuft:

```
victor@hugo:~$ telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 hugo ESMTP Exim 4.44 Thu, 24 Feb 2005 21:16:02 +0100
quit
221 hugo closing connection
Connection closed by foreign host.
victor@hugo:~$
```

Hier meldet sich der Mailserver Exim und wartet auf eine Eingabe. Mit `quit` können Sie die Verbindung wieder schließen. Bricht `telnet` die Verbindung mit einem Fehler wie

```
Trying 127.0.0.1...
```

```
telnet: Unable to connect to remote host: Connection refused
victor@hugo:~$
```

ab, ist der Mailserver nicht gestartet. Am einfachsten starten Sie solche Server-Prozesse über das jeweilige Init-Script: dies finden Sie im Verzeichnis `/etc/init.d`. Die Startscripts heißen dabei in der Regel so wie die Software bzw. das installierte Softwarepaket. Um den bei Debian verwendeten Mailserver Exim zu starten, würde man folgendes Kommando (mit root-Berechtigung) ausführen:

```
hugo:~# /etc/init.d/exim4 start
Starting MTA: exim4.
hugo:~#
```

Tipp

Auch wenn sich die Konfiguration der von SUSE (Postfix) und Fedora (Sendmail) genutzten Mailserver von der des Exim4 unterscheidet, können Sie diese Tests mit Ihrer Variante ebenfalls durchführen!

Ob die lokale Zustellung funktioniert, können Sie mit dem kommandozeilenorientierten Tool `mail` testen.

Wenn Sie `mail` aufrufen und keine E-Mail für Sie vorhanden ist, wird das Tool nur den Kommentar ausgeben »No mail for victor«.

```
victor@hugo:~$ mail
No mail for victor
victor@hugo:~$
```

Um eine E-Mail zu versenden, müssen Sie dem Programm `mail` als letzte Option eine Mailadresse mitgeben. Mit der Option `-s 'Ein Subject'` können Sie zusätzlich einen Betreff setzen.

```
victor@hugo:~$ mail -s 'Ein Subject' victor
```

Danach können Sie den Mailtext schreiben. Wir haben uns in unserem Beispiel auf ein einfaches »Huhu!« beschränkt :-).

```
Huhu! CTRL + D
Cc: Return
```

Mit CTRL + D beenden Sie die Eingabe. Die Angabe einer Mailadresse, an die eine Kopie geschickt werden soll, übergehen Sie, indem Sie einfach die Taste Return drücken. Jetzt können Sie `mail` noch einmal aufrufen. Da jetzt eine E-Mail vorhanden

ist, geht das Programm in den Verwaltungs-Modus und zeigt Ihnen einige Statusinformationen wie z. B. den Titel der neuen E-Mail an. Danach meldet es sich mit einem & als Prompt, an dem Sie Kommandos eingeben können.

```
victor@hugo:~$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/victor": 1 message 1 new
>N  1 victor@hugo  Thu Feb 24 21:18   15/445   Ein Subject
&
```

Ein Druck auf die Taste Return zeigt den Inhalt an.

```
& Return
Message 1:
From victor@hugo Thu Feb 24 21:18:27 2005
Envelope-to: victor@hugo
Delivery-date: Thu, 24 Feb 2005 21:18:27 +0100
To: victor@hugo
Subject: Ein Subject
From: Victor Y. Secosbosque <victor@hugo>
Date: Thu, 24 Feb 2005 21:18:27 +0100

Huhu!
```

Mit D + Return löschen Sie die aktuelle E-Mail, und mit Q + Return beenden Sie das Programm `mail` wieder.

```
& d Return
& q Return
victor@hugo:~$
```

Debian	Das obige Beispiel mit dem Programm `mail` funktioniert auch, wenn Exim4 nicht gestartet ist. Im Prinzip muss auf einem Debian-System die Mailserver-Software nicht als Dämon laufen – es reicht, wenn sie installiert ist.
SUSE	Der Postfix-Dämon muss laufen, um lokale E-Mail abliefern zu können. Wenn Sie nach dem obigen Beispiel eine E-Mail erzeugen, wird diese bei nicht laufendem Postfix in eine Warteschlange gestellt, die abgearbeitet wird, sobald der Dämon gestartet wird. Es geht in diesem Fall keine E-Mail verloren.
Fedora	Unserer obiger Mailtest zeigt bei Fedora ähnliche Ergebnisse wie bei SUSE: Die Software `mail` versucht, die E-Mail über eine Verbindung zur Adresse `localhost` abzuliefern. Wenn der Sendmail-Dämon nicht läuft, scheitert die Ablieferung: Die E-Mail wird in einer separaten Warteschlange gespeichert und nach dem Start des Sendmail-Dämons ausgeliefert.

Tabelle 8.2: Unterschiede zwischen den Distributionen

Damit haben wir von der Anwendungsseite getestet, ob der Mailserver zufriedenstellend funktioniert. Wenden wir uns jetzt etwas elaborierteren administrativen Aufgaben zu.

8.6 Grundlagen der Mailserver-Administration

Ist der Mailserver einmal aufgesetzt und mit den notwendigen Funktionen versehen, ist im laufenden Betrieb i. d. R. kaum ein Eingreifen notwendig. Im Laufe der Zeit müssen vielleicht auf Servern, die multiple Adressen bedienen, die eine oder andere Ergänzung eingetragen werden, oder es ergeben sich Änderungen in den Alias-Einträgen, was häufig bei der Einrichtung neuer Mailinglisten auftritt. Aber auch hier bietet Exim4 Features, die ein manuelles Eingreifen verzichtbar machen: Wenn Sie Mailman für die Listenverwaltung nutzen, existiert die Möglichkeit, diesen ohne Umweg über die Aliases-Datei direkt als Router in der Konfiguration einzutragen.

Debian-Tipp

Wir haben für unsere Beispiele oben bei der Konfiguration von Exim4 die Option ausgewählt, *eine große* Konfigurationsdatei zu nutzen. Wenn Sie zusätzliche Software benutzen, die mit Exim4 zusammenarbeiten soll, lohnt es sich jedoch, auf *mehrere kleine* Konfigurationsdateien umzustellen. In diesem Fall werden die zusätzlichen Konfigurationsschnipsel, die z. B. die Paketbetreuer von Mailman ihrem Paket beigefügt haben, automatisch eingebunden.

Es bleiben im Prinzip zwei Aufgaben übrig: das Management der Warteschlange, falls dort einmal E-Mails aufgelaufen sein sollten, die nicht zustellbar sind, und die Auswertung der Logdateien, um einen Überblick über die Funktionsfähigkeit unseres Mailservers zu bekommen. Um die Warteschlangenverwaltung testen zu können, zeigen wir Ihnen vorher noch einen Trick, um Nachrichten vor der Zustellung einzufrieren.

8.6.1 Einen einfachen systemweiten Filter einrichten

Exim4 besitzt eine eigene Filtersprache, um den Weg einer E-Mail verändern zu können. Im Prinzip ist die Einrichtung einer eigenen Software für das Filtern (wie Procmail oder Maildrop) nicht notwendig. Neben Filtern, die jeder Nutzer in seinem Home-Verzeichnis in einer `.forward`-Datei einrichten kann, besteht die Möglichkeit, einen systemweiten Filter einzurichten. Wir wollen jetzt einmal alle E-Mails von root einfrieren, damit in der Mail-Queue etwas hängen bleibt, was wir dann im folgenden Teil weiterverarbeiten können.

Für den systemweiten Filter existiert die Konfigurationsoption `system_filter`, mit der Sie eine Datei angeben können, die die Filterkommandos enthält. Wir werden später

die Datei `/etc/exim4/exim4.filter` dafür anlegen. Suchen Sie sich jetzt in der Exim4-Konfigurationsdatei die Zeile `begin acl`, und tragen Sie darüber folgende Zeile ein:

Listing 8.1: /etc/exim4/exim4.conf

```
system_filter = /etc/exim4/exim4.filter

begin acl
[...]
```

Debian-Tipp

Wir haben auf die Besonderheit der Konfiguration bei Debian ja bereits hingewiesen: Wenn Sie EINE KONFIGURATIONSDATEI bei der Konfiguration ausgewählt haben, müssen Sie die `system_filter`-Direktive in der Datei `/etc/exim4/exim4.conf.template` eintragen. Haben Sie MEHRERE KONFIGURATIONSDATEIEN ausgewählt, können Sie die Zeile am Ende der Datei `/etc/exim4/conf.d/main/02_exim4-config_options` anfügen. In beiden Fällen müssen Sie hinterher das Kommando `invoke-rc.d exim4 restart` aufrufen, damit Ihre Änderungen übernommen werden.

Jetzt sollten wir noch unseren Mailfilter anlegen: Die Filtersprache von Exim4 ist relativ einfach zu verstehen, insbesondere wenn Sie schon ein wenig Programmiererfahrung haben. Eine Übersicht über die zur Verfügung stehenden Variablen und Aktionen bietet die »Exim Filter Specification«, die bei der Dokumentation zu Exim4 enthalten ist.

Listing 8.2: /etc/exim4/exim4.filter

```
if $sender_address_local_part is "root“ and first_delivery then
   freeze
endif
```

In unserem Filter fragen wir ab, ob root der Versender der E-Mail ist und ob es sich um den ersten Versuch der Auslieferung handelt. In diesem Fall frieren wir die E-Mail mit dem Kommando `freeze` ein. Andere mögliche Aktionen wären `save $home/irgend/eine/datei`, um die E-Mail zu speichern, oder `deliver irgendwer@irgendwo`, um die E-Mail weiterzuleiten.

Fehler im Systemfilter werden in der Logdatei `/var/log/exim4/paniclog` gemeldet. Wenn Sie sich z. B. bei der Angabe der Variable `sender_address_local_part` verschrieben haben, finden Sie im `paniclog` den Eintrag:

```
[...] Error in system filter: failed to expand "$sender_adress_local_part“
      in filter file [...]
```

Ob unser Filter funktioniert, testen wir auf bewährte Weise mit Hilfe von `mail`. Beachten Sie, dass der Absender immer die Kennung ist, unter der Sie gerade agieren!

Wir haben das Einfrieren für die root-Kennung eingerichtet, also müssen wir unsere Testmail auch als root verschicken:

```
hugo:~# mail -s 'Test' victor@localhost
Huhu!
Cc:
hugo:~# mailq
 5m   310 1Faalq-0002Zk-53 <root@hugo>
          victor@localhost

hugo:~#
```

Aber wir greifen vor, denn das `mailq`-Kommando kommt erst im folgenden Abschnitt.

8.6.2 Der Kampf mit der Queue

Mail-Queues sind ein Problem, wenn sie nicht so funktionieren, wie sie sollen. Das passiert vor allem dann, wenn Exim4 E-Mails nicht zustellen kann, aber der Ansicht ist, es gäbe später noch Gelegenheit, dies zu tun. In den meisten Fällen führen Fehler beim Versenden direkt zu einer Rückantwort (engl. »Bounce«) mit einem Hinweis, warum die Zustellung schiefgegangen ist. Kommt es vor, dass E-Mails in der Queue hängen bleiben, existieren einige Optionen für das Kommando `exim`, um einen erneuten Versuch der Auslieferung anzustoßen oder die E-Mails zu löschen.

Beginnen wir mit der Ausgabe der Queue; wenn Sie, wie im vorherigen Abschnitt gezeigt, E-Mails von root einfrieren und den letzten Test bereits durchgeführt haben, sollte Ihnen die folgende Ausgabe bekannt vorkommen. Aber eventuell haben Sie ja auch Pech und in Ihrer Queue hängen regulär auszuliefernde E-Mails

```
hugo:~# mailq
 5m   310 1Faalq-0002Zk-53 <root@hugo>
          victor@localhost

hugo:~# exim -Mvl 1Faalq-0002Zk-53
200n-nn-nn 18:18:59 Received from root@hugo U=root P=local S=306
*** Frozen by the system filter
hugo:~# exim -Mvh 1Faalq-0002Zk-53
1Faalq-0002Zk-53
root 0 0
<root@hugo>
1146500339 0
-ident root
-received_protocol local
-body_linecount 1
-auth_id root
```

```
-auth_sender root@Theodor
-allow_unqualified_recipient
-allow_unqualified_sender
-frozen 1146500339
-local
XX
1
victor@hugo

161P Received: from root by hugo with local (Exim 4.61)
        (envelope-from <root@hugo>)
        id 1Faalq-0002Zk-53
        for victor@hugo; Mon, 01 May 2006 18:18:59 +0200
012* To: victor
020T To: victor@hugo
014  Subject: Test
040I Message-Id: <E1Fab6x-0002fy-Kv@Theodor>
026F From: root <root@hugo>
038  Date: Ttt, dd Mmm 200n 18:18:59 +0200
hugo:~#
```

Das Kommando `mailq` liefert eine Übersicht über alle zurzeit eingefrorenen E-Mails. Wichtig für die weitere Verarbeitung ist dabei die ID der E-Mail, in unserem Beispiel `1Faalq-0002Zk-53`. Diese ID benötigen wir für die weitere Verarbeitung.

Wir haben jetzt einerseits die Möglichkeit, Informationen über den Status der in der Queue hängenden E-Mail auszugeben. Der Aufruf von `exim -Mvl` liefert uns einen Hinweis, warum diese E-Mail hängen geblieben ist. Mit `exim -Mvh` erhalten Sie eine – etwas unübersichtliche – Ausgabe der Header der E-Mail inklusive einiger zusätzlicher, serverinterner Informationen. Für unsere Zwecke wichtig sind die `To:`- und `From:`-Zeilen, da wir diesen entnehmen können, von wem und an wen die E-Mail gehen sollte. Sie bilden die Basis für die Entscheidung darüber, was als Nächstes zu tun ist.

Was wir als Nächstes tun können und müssen, hängt auch von dem Grund ab, warum die E-Mails hängen:

- Wir haben z. B. einen Ausfall des Netzwerks gehabt. Jetzt funktioniert es wieder und wir wollen alle hängen gebliebenen E-Mails zustellen. Mit dem Kommando `exim -qff` sorgen wir dafür, dass für *alle* hängen gebliebenen E-Mails ein neuer Auslieferungsversuch durchgeführt wird. Wenn die Queue zu groß geworden ist und Sie die Netzwerkverbindung nicht überlasten wollen, können Sie auch mit `exim -qf` versuchen, zuerst alle nicht-eingefrorenen E-Mails auszuliefern.
- Wenn wir explizit nur eine eingefrorene E-Mail erneut zustellen lassen wollen, können wir dies mit `exim -M 1Faalq-0002Zk-53` erreichen. Sie können hinter der Option `-M` auch mehrere IDs angeben.

- Wenn wir – auf anderem Weg – herausbekommen haben, dass diese E-Mail nicht oder nicht mehr zustellbar ist, können wir mit `exim -Mg 1Faalq-0002Zk-53` den Auslieferungsvorgang abbrechen. Der Absender erhält dann eine Nachricht, dass das Versenden der E-Mail nicht funktioniert hat.
- Für den Fall, dass uns bekannt ist, dass diese E-Mail weder vom Empfänger benötigt wird noch der Absender eine Nachricht über die fehlerhafte Zustellung bekommen muss, können wir sie mit `exim -Mrm 1Faalq-0002Zk-53` komplett aus der Queue löschen.

Damit kennen Sie die grundlegenden Kommandos zur Bearbeitung der Mail-Queue. Bleibt noch die Frage, warum die Optionen so unglaublich intuitiv ausgefallen sind: Vor der Entwicklung des »Mailserver sapiens« existierte ein Dinosaurier namens Sendmail, der so weit verbreitet war, dass der Umgang mit ihm zum Standard für alle nachfolgenden Mailserver wurde. Das heißt z. B., dass Sie immer noch ein Programm `sendmail` auf Ihrem Server finden werden, auch wenn Sie Sendmail gar nicht mehr installiert haben – es ist meist ein Link auf die eigentlich installierte Software. Viele Programme, die intern E-Mails versenden, erwarten nämlich ein Kommando namens `sendmail`. Und damit dieses Kommando dann auch »abwärtskompatibel« ist, müssen natürlich auch die Optionen denen des Originals entsprechen. Und so haben sich die wundersamen Buchstabenkombinationen bis in die heutige Zeit gerettet und werden uns erhalten bleiben bis ans Ende aller Software.

8.6.3 Auswertung der Log-Dateien mit eximstats

Auch wenn die Auswertung der Mail-Logs nicht die Bedeutung hat, die im Allgemeinen Webstatistiken zukommt, sollte man doch eine Analyse des Mailverkehrs nicht gering schätzen. Mit eximstats liegt Exim4 ein Perl-Script bei, mit dem sich die Log-Dateien auswerten lassen:

```
hugo:~# eximstats -txt /var/log/exim4/mainlog /var/log/exim4/mainlog.1

Exim statistics from 200n-nn-nn 14:00:50 to 200n-nn-nn 16:57:11

Grand total summary
-------------------
                                                  At least one address
  TOTAL               Volume    Messages     Hosts      Delayed       Failed
  Received             141MB       27789         2       0  0.0%      3  0.0%
  Delivered            141MB       27786         1

Deliveries by transport
-----------------------
                      Volume    Messages
  mail_spool           137MB       27090
  procmail_pipe       3819KB         696
```

```
Messages received per hour (each dot is 84 messages)
----------------------------------------------------

[...]
06-07     83
07-08    137 .
08-09    249 ..
09-10   1900 ......................
10-11    854 ..........
11-12   1547 ..................
[...]

Deliveries per hour (each dot is 93 deliveries)
-----------------------------------------------

[...]
06-07     10
07-08     75
08-09    196 ..
09-10   1394 ..............
10-11   1307 ..............
11-12   1007 ..........
[...]

Time spent on the queue: all messages
-------------------------------------

Under   1m    12238  44.0%   44.0%
        5m       40   0.1%   44.2%
       15m      184   0.7%   44.8%
       30m    12874  46.3%   91.2%
        1h       52   0.2%   91.4%
        3h      432   1.6%   92.9%
        6h      384   1.4%   94.3%
       12h      290   1.0%   95.3%
        1d     1006   3.6%   99.0%
Over    1d      290   1.0%  100.0%

Top 50 local senders by message count
-------------------------------------

     24       12KB   root
      4       3830   Debian-exim
      1        333   victor
[...]
```

```
Top 50 local destinations by message count
-----------------------------------------

  27785      141MB   victor

[...]

List of errors
--------------

    1 asdfasdf@hugo: Unrouteable address

    1 gutmann@lena.mmz.uni-duesseldorf.de R=nonlocal: Mailing
            to remote domains not supported

    1 victor@antonius R=nonlocal: Mailing to remote domains
            not supported

Errors encountered: 3
---------------------
hugo:~#
```

An der (gekürzten) Ausgabe sind neben den Zusammenfassungen insbesondere interessant:

- Plötzliche Häufungen von Fehlern sollten näher untersucht werden. Die Fehler in unserem Beispiel lassen sich leicht erklären, da sie auf Fehlversuchen basieren: es gab eine falsche E-Mail-Adresse und zwei Versuche, mit Hilfe von Exim4 E-Mails ins Internet zu versenden, was in der Konfiguration deaktiviert ist.
- Die `Deliveries by transport` lassen je nach Ergänzung eine Beurteilung der Verarbeitung nach dem Empfang zu. Hier können Sie z. B. erkennen, welche direkt an Exim4 angebundenen Mailfilter von Ihren Usern genutzt werden.
- Dass die Anzahl der eingehenden und ausgehenden E-Mails – bis auf die drei fehlgeleiteten E-Mails – identisch ist, liegt an der speziellen Konfiguration des Beispielservers. Hier ist Exim4 nur für die Verteilung der über Fetchmail eingehenden E-Mails zuständig. Bei einem regulären Mailserver können Sie hier erkennen, wie viele E-Mails hereinkommen bzw. hinausgehen.

Anstelle der etwas »drögen« Textausgabe können Sie `eximstats` auch zu einer HTML-Ausgabe bewegen. Haben Sie die GD-Bibliotheken für Perl installiert, kann `eximstats` mit der Option `-charts` auch eine grafische Ausgabe erzeugen. Um regelmäßig Statistiken zu erzeugen, empfehlen wir, `eximstats` in das `logrotate`-Script einzubinden.

8.7 POP3- und IMAP-Server einrichten

Da Mailserver und Mailclient oft nicht auf dem gleichen physikalischen Server laufen, wurden Protokolle entwickelt, die den Abruf von E-Mail von einem Server regeln. Das ältere von beiden, POP3, das **P**ost **O**ffice **P**rotocol in der Version 3, ermöglicht den Zugriff auf die serverseitig gespeicherte Eingangsmailbox und ist hauptsächlich für den Abruf der darin enthaltenen E-Mails gedacht.

Die modernere Variante IMAP, das **I**nternet **M**essage **A**ccess **P**rotocol, bietet neben dem reinen Abruf auch die Möglichkeit, auf dem Server Ordner und Unterordner anzulegen. Damit kann ein IMAP-Server so genutzt werden wie ein lokaler Mailclient, mit dem zusätzlichen Nutzen, von einem speziellen PC unabhängig zu sein und im Prinzip von jedem PC aus auf alle eigenen E-Mails zugreifen zu können.

Was ist nun notwendig, um den Nutzern den Abruf der E-Mails per POP3 oder IMAP zu ermöglichen? Der Server muss zumindest in der Lage sein, E-Mails für die Nutzer annehmen zu können, die diese abrufen wollen:

- Der Mailserver muss so eingerichtet sein, dass er E-Mails annehmen und lokal ausliefern kann.
- Es müssen (System-)Kennungen für die Nutzer eingerichtet sein, die E-Mails empfangen und abrufen können sollen.

8.7.1 Dovecot: Kombiserver für POP3 und IMAP

Um uns und Ihnen die Arbeit zu erleichtern, haben wir uns für den Einsatz von Dovecot entschieden. Diese Software beherrscht neben POP3 und IMAP auch deren SSL-verschlüsselte Varianten und ist einfach zu installieren und zu warten. Sie benötigen für den Einsatz von Dovecot:

Installation eines Mailservers	Debian	Fedora	SUSE
Basispaket Das Basispaket für die Dovecot-Installation wird i. d. R. bei der Installation einer der Serverkomponenten mit installiert	dovecot-common	—	—
POP3-Server Die POP3-Serverkomponente	dovecot-pop3d	dovecot	dovecot
IMAP-Server Die IMAP-Serverkomponente	dovecot-imapd	dovecot	dovecot
Nutzerverwaltung Für die Anbindung an die verschiedenen Möglichkeiten der Nutzerverwaltung wie LDAP, MySQL oder PostgreSQL sind zusätzliche Bibliotheken notwendig	libldap2 libmysqlclient2 libpq3	—	in dovecot enthalten
SSL-Verschlüsselung Um SSL-gesicherte Verbindungen mit Dovecot nutzen zu können muß openssl installiert sein	openssl libsslx.x.x	in dovecot enthalten	in dovecot enthalten

Tabelle 8.3: Paketübersicht

Nach der Installation der Softwarepakete finden Sie entweder in `/etc/dovecot` oder in `/usr/share/doc/dovecot/examples` die Datei `dovecot-example.conf`. Diese Datei müssen Sie nach `/etc/dovecot/dovecot.conf` kopieren und dort ein paar Anpassungen vornehmen.

Debian-Tipp

Für Debian existiert ein Dummy-Paket `dovecot`, das automatisch alle benötigten Komponenten für einen POP3- und einen IMAP-Server installiert. Außerdem wird bei der Installation automatisch ein SSL-Zertifikat angelegt und in `/etc/ssl/...` gespeichert – achten Sie bei der Ausgabe der Konfiguration darauf. Außerdem wird eine Variante der `dovecot-example.conf` als `/etc/dovecot/dovecot.conf` angelegt, sodass Sie keine Kopie machen müssen.

Die Beispielkonfiguration ist so gestaltet, dass auch ein irrtümlich gestarteter Dovecot keinen Schaden anrichten kann bzw. i. d. R. gar nichts tut. Gehen wir kurz die wichtigsten Optionen durch, an denen Sie Veränderungen vornehmen müssen, um Dovecot lauffähig zu machen:

Listing 8.3: /etc/dovecot/dovecot.conf

```
# If you're in a hurry, see http://wiki.dovecot.org/QuickConfiguration

# '#' character and everything after it is treated as comments. Extra spaces
# and tabs are ignored. If you want to use either of these explicitly, put the
# value inside quotes, eg.: key = "# char and trailing whitespace  "

# Default values are shown after each value, it's not required to uncomment
# any of the lines. Exception to this are paths, they're just examples
# with real defaults being based on configure options. The paths listed here
# are for configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var
# --with-ssldir=/etc/ssl
```

Die ersten Kommentare geben uns einen Hinweis darauf, wie wir mit der Konfigurationsdatei umzugehen haben:

- Kommentare beginnen mit einem #.
- Die in auskommentierten Zeilen angegebenen Werte stellen die Vorgabe dar. Wir müssen also keine dieser Zeilen einkommentieren, wenn wir den Standardwert beibehalten wollen.
- Nur bei Pfaden funktioniert dies etwas anders. Da sollten wir uns aber auf unsere Distributoren verlassen können: Diese haben i. d. R. die Pfade beim Kompilieren so konfiguriert, wie es in der uns vorliegenden Beispielkonfiguration angegeben ist, sodass wir auch hier Zeilen nur auskommentieren müssen, wenn wir die Vorgaben ändern wollen.

Listing 8.4: /etc/dovecot/dovecot.conf

```
[...]

# Protocols we want to be serving:
#  imap imaps pop3 pop3s
protocols = imap imaps pop3 pop3s
```

Die nächsten Zeilen können wir links liegen lassen; die erste wichtige – aber auch wirklich wichtige – Zeile beginnt mit dem Schlüsselwort `protocols`. Werden hier keine Protokolle angegeben, startet Dovecot gar nicht erst. Um Dovecot zur Arbeit zu bewegen, sollte dort zumindest ein Protokoll eingetragen sein; wir tragen dort einmal alle vier möglichen Protokolle ein. Es tut aber auch ein Subset wie z. B. `pop3` und `pop3s`, wenn Sie sich vorerst die Gedanken über den für IMAP notwendigen Speicherplatz sparen wollen.

Listing 8.5: /etc/dovecot/dovecot.conf

```
# IP or host address where to listen in for connections. It's not currently
# possible to specify multiple addresses. "*" listens in all IPv4 interfaces.
# "[::]" listens in all IPv6 interfaces, but may also listen in all IPv4
# interfaces depending on the operating system.
#
# If you want to specify ports for each service, you will need to configure
# these settings inside the protocol imap/pop3 { ... } section, so you can
# specify different ports for IMAP/POP3. For example:
#   protocol imap {
#     listen = *:10143
#     ssl_listen = *:10943
#     ..
#   }
#   protocol pop3 {
#     listen = *:10100
#     ..
#   }
#listen = *

# IP or host address where to listen in for SSL connections. Defaults
# to above if not specified.
#ssl_listen =
```

Haben Sie einen Server mit mehreren Netzwerkkarten oder mit mehreren IP-Adressen laufen, der eventuell als Firewall oder Gateway arbeitet, können Sie hier bestimmen, auf welchen IP-Adressen Dovecot auf Anfragen lauschen soll. Bei einem Linux-Router wäre es z. B. sinnvoll, mit IMAP und POP3 nur auf dem Interface zu antworten, das zum lokalen Netzwerk gehört. Die SSL-verschlüsselte Variante

könnte aber z. B. auch nach draußen angeboten werden. Einträge dafür könnten beispielsweise so aussehen: `listen = 192.168.1.10:*` und `ssl_listen = *` .

Listing 8.6: /etc/dovecot/dovecot.conf

```
[...]

# Disable SSL/TLS support.
#ssl_disable = no

# PEM encoded X.509 SSL/TLS certificate and private key. They're opened before
# dropping root privileges, so keep the key file unreadable by anyone but
# root.
#ssl_cert_file = /etc/ssl/certs/dovecot.pem
#ssl_key_file = /etc/ssl/private/dovecot.pem
```

Die Möglichkeit, SSL-verschlüsselte Verbindungen zu nutzen, ist bei Dovecot generell angeschaltet. Da bei der Installation Zertifikate generiert werden, funktionieren die SSL-Varianten von IMAP und POP3 ohne weiteres Zutun Ihrerseits. Wenn Sie Dovecot mit Zertifikaten von einer externen Certificate Authority nachrüsten wollen, können Sie entweder die automatisch generierten ersetzen oder die Pfade in `ssl_cert_file` und `ssl_key_file` anpassen.

Listing 8.7: /etc/dovecot/dovecot.conf

```
# Disable LOGIN command and all other plaintext authentications unless
# SSL/TLS is used (LOGINDISABLED capability). Note that 127.*.*.* and
# IPv6 ::1 addresses are considered secure, this setting has no effect if
# you connect from those addresses.
#disable_plaintext_auth = yes
```

Überspringen wir ein paar Zeilen, und widmen uns kurz noch der Eintragung `disable_plaintext_auth` . Die Übermittlung von Passwörtern im Klartext ist standardmäßig für die Zugriffe abgeschaltet, die als unsicher gelten, also diejenigen über `POP3` und `IMAP`. Das heißt nicht, dass kein Zugriff möglich ist, sondern nur, dass die Clients die Passwörter vor dem Übermitteln verschlüsseln müssen. Beachten Sie, dass die lokalen Loopback-Schnittstellen als sicher eingestuft werden, da die versendeten IP-Pakete hier nur serverintern hin und her wandern und i. d. R. nicht belauscht werden können!

8.7.2 Mailbox-Dateien und das Problem der vielen Köche

Beim Umgang mit Mailboxen haben wir generell das Problem, dass häufig mehrere Prozesse auf die gleichen Dateien zugreifen. Damit es hier nicht zu datentechnischen Verkehrsunfällen kommt, müssen Verfahren eingesetzt werden, die den

gleichzeitigen Zugriff unterbinden. In zwei Bereichen müssen wir uns über dieses Problem Gedanken machen:

- Bei den Eingangsmailboxen der Nutzer: Diese werden in unserem Beispiel vom Mailserver Exim4 geschrieben und von Dovecot zum Lesen und – nicht zu vergessen – zum Löschen geöffnet.
- Wenn wir Dovecot als IMAP-Server einsetzen, benötigt dieser einen Bereich, in dem er Unterordner anlegen und E-Mails speichern kann. Dies ist i. d. R. ein Unterordner des Home-Verzeichnisses.

Um das erste Problem in den Griff zu bekommen, existiert unter Linux die Systemfunktion `fcntl`, die verhindert, dass zwei Prozesse gleichzeitig eine Datei öffnen. Normalerweise müssen Sie sich keine Gedanken über dieses Problem machen; es gibt jedoch Ausnahmen: Sie befinden sich nicht auf einem Linux-System oder das Dateisystem, auf dem die Mailboxen gelagert werden, wird per NFS eingebunden. In letzterem Fall können Sie mit Hilfe des lockd-Dämons `fcntl` auch mit NFS zum Laufen bringen. Oder Sie greifen auf die sogenannte »dotlock«-Methode zurück; bei dieser Methode werden Lock-Dateien angelegt, die den anderen Prozessen anzeigen, dass bereits ein Prozess auf die Datei zugreift. Dies funktioniert natürlich nur, wenn alle Prozesse, die mit den Mailboxen arbeiten, auf dieses Verfahren umgestellt werden. In unserem Beispiel wären das zumindest Exim4 und Dovecot selbst.

Listing 8.8: /etc/dovecot/dovecot.conf

```
[...]

# Grant access to these extra groups for mail processes. Typical use would be
# to give "mail" group write access to /var/mail to be able to create dotlocks.
mail_extra_groups = mail

[...]

# Don't use mmap() at all. This is required if you store indexes to shared
# filesystems (NFS or clustered filesystem).
#mmap_disable = no

[...]

# Locking method for index files. Alternatives are fcntl, flock and dotlock.
# Dotlocking uses some tricks which may create more disk I/O than other locking
# methods. NOTE: If you use NFS, remember to change also mmap_disable setting!
#lock_method = fcntl
```

Für Dovecot müssen dann die Zeilen `lock_method` auf `dotlock` und zusätzlich `mmap_disable = yes` gesetzt sein. Außerdem benötigt Dovecot für die Erstellung der Lock-

Dateien Schreibrechte im Mail-Verzeichnis. Dies wird i. d. R. über eine spezielle Gruppe geregelt, die Schreibrechte auf das System-Mailverzeichnis hat und unter der Dovecot laufen muss. Sie können diese in `mail_extra_groups` angeben.

Das zweite Problem ist ein wenig gravierender: Mailclients nutzen i. d. R. das Verzeichnis `$HOME/Mail`, um dort die lokal verwalteten E-Mails zu speichern. Auch wenn für die Mailbox-Dateien von allen Clients die gleichen Formate benutzt werden, sorgen unterschiedliche Arten der Index-Erstellung und der Benennung von Unterverzeichnissen dafür, dass sich verschiedene Mailclients i. d. R. ein Mailverzeichnis *nicht* teilen können. Dies gilt auch für Dovecot, das ebenfalls in der IMAP-Standardkonfiguration das Verzeichnis `$HOME/Mail` nutzen möchte. Solange auf dem Server jedoch kein lokaler Mailclient genutzt wird, treten keine Probleme auf. Wenn Sie jedoch die Funktion von Dovecot auf Ihrer Linux-Workstation testen wollen und gleichzeitig unter Ihrer Kennung einen Mailclient benutzen, sollten Sie Dovecot überreden, einen anderen Ordner in Ihrem Homeverzeichnis für die Speicherung der IMAP-Ordner zu benutzen:

Listing 8.9: /etc/dovecot/dovecot.conf

```
[...]

# Default MAIL environment to use when it's not set. By leaving this empty
# dovecot tries to do some automatic detection as described in
# /usr/share/doc/dovecot-common/mail-storages.txt. There's a few special
# variables you can use, eg.:
#
#   %u - username
#   %n - user part in user@domain, same as %u if there's no domain
#   %d - domain part in user@domain, empty if there's no domain
#   %h - home directory
#
# See /usr/share/doc/dovecot-common/variables.txt for full list. Some examples:
#
#   default_mail_env = maildir:/var/mail/%1u/%u/Maildir
#   default_mail_env = mbox:~/mail/:INBOX=/var/mail/%u
#   default_mail_env = mbox:/var/mail/%d/%n/:INDEX=/var/indexes/%d/%n
#
default_mail_env = mbox:%h/DoveMail/:INBOX=/var/mail/%u
```

Die Direktive `default_mail_env` enthält Angaben, nach denen Dovecot E-Mails empfängt und speichert. Um eine Überschneidung mit eventuell genutzten Mailclients zu verhindern, haben wir mit `mbox:%h/DoveMail` den Speicherort für IMAP-Ordner auf `DoveMail` im Homeverzeichnis (`%h`) gesetzt. Die Ergänzung `INBOX=/var/mail/%u` sorgt dafür, dass Dovecot eingehende E-Mail weiterhin an der richtigen Stelle sucht und nicht etwa im Homeverzeichnis des Nutzers.

Achtung

Bei Debian-Systemen kann es vorkommen, dass Dovecot in der Standardkonfiguration die eingehenden E-Mails nicht findet! Ergänzen Sie dann die nicht ausgefüllte Konfiguration `default_mail_env` durch `mbox:/var/mail/%u`.

8.8 Gibt es Mittel gegen Spam und Viren?

Zu den Gefahren, die mit der Nutzung von E-Mail einhergehen, gehören heutzutage

- das Einschleppen von Viren durch als Anhang verschickte Dateien. Mit einem Virenscanner, der die Anhänge einer E-Mail vor der Speicherung in das Postfach auf Viren kontrolliert, lassen sich diese Probleme aber größtenteils in den Griff bekommen; insbesondere, wenn die Signaturen automatisch aktualisiert werden können.
- unerwünschte E-Mails mit bösartigen Absichten. Dazu gehören hauptsächlich Phishing-Mails und Versuche, die Empfänger auf Webseiten zu locken, die versuchen, durch eingebetteten Schadcode Viren oder Ähnliches auf dem Rechner des Empfängers zu installieren. Diese Probleme können entweder durch zusätzliche Funktionen des Virenscanners oder durch Spam-Filter erkannt und beseitigt werden.
- Spam-E-Mails im Allgemeinen. Diese stellen i. d. R. keine Gefahr mehr dar, sind aber lästig und beeinträchtigen den Mailverkehr insgesamt. Hierfür existieren verschiedene Lösungswege: Einerseits werden die E-Mails selbst darauf untersucht, ob sie Spam enthalten, oder es werden Listen von Mailservern oder Absendern gepflegt, die als Versender von Spam bekannt sind.

Zäumen wir das Pferd einmal von hinten auf und beginnen wir mit dem »Lästigen«, dem Spam:

- Eine Möglichkeit ist das Führen schwarzer und weißer Listen von Mail-Hosts, von denen E-Mails abgelehnt bzw. angenommen werden. Dies können Sie entweder selbst tun oder durch sogenannte RBL- bzw. DNSRBL-Server erledigen lassen.
- Eine weitere Möglichkeit ist das Führen schwarzer und weißer Listen von Absendern, was oft als Ergänzung zu Listen von Mail-Hosts genutzt wird. Hier sind insbesondere die weißen Listen von Bedeutung, in denen E-Mail-Adressen von Personen eingetragen werden, die auf keinen Fall vom Spam-Filter aussortiert werden sollen.
- Zuletzt können Sie Spam-Filter einsetzen, die versuchen, anhand der E-Mail herauszufinden, ob es sich dabei um Spam oder eine reguläre E-Mail handelt. Hier existiert einerseits (meist von kommerziellen Anbietern geführte) Software,

die regelmäßig mit Spam-Mail-Signaturen gefüttert wird und so die aktuell im Umlauf befindlichen Spam-Mails herauszufiltern versucht. Ein anderer Weg ist der Einsatz von Spam-Filtern auf dem Rechner des Empfängers. Diese Filter können auf den spezifischen E-Mail-Verkehr des Anwenders angelernt werden und so für eine hohe Trefferquote ohne falsch positive Meldungen sorgen.

8.8.1 Von schwarzen und weißen Listen

Eine Möglichkeit, sich vor unerwünschter E-Mail zu schützen, besteht darin, sich die Server, von denen man E-Mails annimmt, genauer auszusuchen – und natürlich darin, dem eigenen Mailserver den Umgang mit den »bösen« Mailservern zu verbieten. Wir lassen unsere Töchter ja auch nicht mit jedem ausgehen. Mittlerweile bringen alle Mailserver Funktionen mit, die bei der Annahme von E-Mails nach Hosts oder Absenderadressen differenzieren können.

Die Konfigurationsdateien von Exim4 enthalten bereits Verweise auf Dateien, in denen Sie definieren können, von welchen Hosts (`local_host_whitelist`) oder Absendern (`local_sender_whitelist`) E-Mails unbedingt *angenommen* oder von welchen Hosts (`local_host_blacklist`) oder Absendern (`local_sender_blacklist`) E-Mails *abgelehnt* werden sollen. Diese Dateien existieren i. d. R. noch nicht und müssen von Ihnen im Konfigurationsverzeichnis von Exim4 angelegt werden (bei Debian-Systemen wäre das `/etc/exim4`).

Die wichtigen Teile der Konfigurationsdatei für die Auswertung schwarzer oder weißer Listen finden sich hinter der Direktive `begin acl` in der Exim4-Konfigurationsdatei.

Debian-Tipp

Je nach Installation finden Sie die Anweisungen für die Konfiguration der schwarzen und weißen Listen in der Datei `/etc/exim4/exim4.conf.template`, wenn Sie bei der Installation EINE KONFIGURATIONSDATEI ausgewählt haben, oder in den Dateischnipseln im Verzeichnis `/etc/exim4/conf.d/acl`, wenn Sie VIELE KLEINE DATEIEN angegeben haben.

```
begin acl
[...]
acl_whitelist_local_deny:
  accept
    hosts = ${if exists{CONFDIR/local_host_whitelist}\
                {CONFDIR/local_host_whitelist}\
                {}}
  accept
    senders = ${if exists{CONFDIR/local_sender_whitelist}\
                  {CONFDIR/local_sender_whitelist}\
                  {}}
```

Hier haben wir als Beispiel für Sie den Ausschnitt, mit dem die Einbindung zweier weißer Listen definiert ist. Sie können an den `if`-Statements erkennen, dass die Datei nur ausgewertet wird, wenn sie existiert. Findet Exim4 keine schwarze oder weiße Liste, werden alle E-Mails verarbeitet. `CONFDIR` ist eine Variable, in der das Konfigurationsverzeichnis von Exim4 definiert ist.

Die Verarbeitung einer schwarzen Liste ist vergleichsweise einfach: Sie tragen zeilenweise in die Datei `/etc/exim4/local_host_blacklist` die IP-Adressen oder Namen der Hosts ein, von denen Sie keine E-Mail mehr empfangen möchten:

```
195.37.191.38
*.google.com
134.99.0.0/16
```

Sie können auf verschiedene Arten auch Gruppen von IP-Adressen oder Namen angeben. Eine genaue Beschreibung der »Schreiberleichterungen« finden Sie in der Exim4-Dokumentation in Kapitel 10.

Etwas schwieriger wird es, wenn Sie weiße Listen einsetzen wollen, da Sie dann zuerst einmal den Zugriff auf einige wenige Domains beschränken müssen, die Sie dann mit der Whitelist erweitern können. Wir zeigen Ihnen ein kurzes Beispiel an unserem Demo-Host:

```
acl_smtp_rcpt = my_acl_check_rcpt

my_acl_check_rcpt:

  accept
    hosts = 127.0.0.1:192.168.1.10

  accept
    acl = acl_whitelist_local_deny

  deny
    message = we accept not everything
```

Die Direktive `acl_smtp_rcpt` zeigt in der Standardkonfiguration auf ein vordefiniertes ACL-Schema mit dem Namen `acl_check_rcpt`. Wir leiten dieses jetzt auf unsere eigene Definition um, die wir `my_acl_check_rcpt` genannt haben. Diese ist sehr rudimentär; wenn Sie sich das Original näher ansehen, werden Sie noch einige Feinheiten feststellen, die Sie vielleicht in Ihre eigene Konfiguration übernehmen sollten. Wir beschränken uns auf zwei wichtige Regeln:

- Die erste Regel besagt, dass die lokale Loopback-Schnittstelle und die lokale IP-Adresse als Versender auftreten dürfen. Sonst würde das lokale Versenden von E-Mails nicht funktionieren.
- Die zweite Regel bezieht sich auf eine Direktive, die bereits in der Konfigurationsdatei enthalten ist und die Dateien für das Whitelisting einbindet. Damit

werden E-Mails von den in der Datei `/etc/exim4/local_host_whitelist` aufgeführten Versendern ebenfalls zugelassen.

- Alles, was nicht in den ersten beiden Regeln als gut befunden wurde, wird in der letzten Regel entsorgt. Die Absender der E-Mails, die wir nicht annehmen, erhalten den Hinweis, dass wir nicht auf alles reagieren, was an unser Fenster klopft.

8.8.2 RBL- und DNSRBL-Server für die Spam-Abwehr

Pflegen Sie Ihre schwarzen Listen selbst, haben Sie die Kontrolle über diejenigen Mails, die nicht angenommen werden sollen. Aus diesem Grund ist Exim4 bei den selbst gepflegten Listen auch so konfiguriert, dass er die »harte Tour« nimmt und E-Mails ablehnt. Die Pflege eigener schwarzer Listen ist jedoch aufwändig; Abhilfe bieten sogenannte RBL- oder DNSRBL-Server, die Ihnen die Pflege der schwarzen Listen abnehmen. Die Betreiber dieser Server sorgen – ähnlich wie bei Suchmaschinen – mit automatisierten Verfahren für den neuesten Stand bezüglich IP-Nummern, die sich des Spam-Versands schuldig gemacht haben.

Die Betreiber der schwarzen Listen testen regelmäßig Server auf ihre Sicherheit. Wird dabei ein Mailserver gefunden, der ein Relaying von E-Mails zulässt, landet dieser automatisch in der Liste der Spam-verdächtigen Server. Die Systemadministratoren werden von diesem Umstand per E-Mail benachrichtigt und müssen jetzt dafür Sorge tragen, dass die Relay-Funktion abgeschaltet oder so konfiguriert wird, dass die Tests der Blacklist-Betreiber nicht mehr greifen. Dann können die Administratoren den Blacklist-Betreiber anweisen, einen weiteren Test durchzuführen. Erst nach einer positiven Testung wird der Mailserver wieder von der schwarzen Liste entfernt. Bis dahin kann er keine E-Mails mehr ausliefern.

Tipp

Wenn Sie sich sorgen, ob Ihr Mailserver auf einem der vielen RBL-Server gelistet sein könnte, können Sie dies über `rbls.org` testen. Da zu den Tests auch die Webadresse des jeweiligen Betreibers angezeigt wird, bietet Ihnen der Check auch eine Übersicht über die aktuell zur Verfügung stehenden RBL- und DNSRBL-Betreiber.

Sie sehen, es ist einfach, auf einer solchen Liste zu landen, aber schwierig, wieder von ihr herunterzukommen. Sorgen Sie daher dafür, dass Ihre Mailserver immer korrekt konfiguriert werden, und achten Sie insbesondere bei der Konfiguration der Relay-Funktion darauf, dass diese nicht über das Internet missbraucht werden kann. Für Sie als Empfänger kann es aber auch bedeuten, dass Ihr Mailserver E-Mails einer für Sie wichtigen Domain ablehnen würde. Daher arbeitet die Einbindung der DNSRBLs in der Default-Konfiguration mit dem sogenannten *Tagging*, kennzeichnet E-Mails also nur als Spam, anstatt sie abzulehnen:

```
[...]
  # Check against classic DNS "black" lists (DNSBLs) which list
  # sender IP addresses
  .ifdef CHECK_RCPT_IP_DNSBLS
  warn
    message = X-Warning: $sender_host_address is listed at $dnslist_domain ($dnslist_value:
                                                                          $dnslist_text)
    log_message = $sender_host_address is listed at $dnslist_domain ($dnslist_value:
                                                                          $dnslist_text)
    dnslists = CHECK_RCPT_IP_DNSBLS
  .endif
[...]
```

Durch das Setzen des Makros `CHECK_RCPT_IP_DNSBLS` wird die angezeigte Direktive aktiviert, die die E-Mail um eine Header-Zeile `X-Warning` ergänzt, falls die IP-Nummer des Absenders vom DNSBLS-Server als Spam-verdächtig markiert wird. Sie können `X-Warning` auch durch eine deutlichere Variante wie `X-Spamverdacht` ersetzen. Wichtig ist nur, dass Sie die Header-Zeile Ihren Mailnutzern mitteilen, damit diese dann lokal danach filtern können.

Tipp

Haben Sie gemerkt, dass die Konfigurationsoption `message` unterschiedlich reagiert, je nachdem, in welchem Zusammenhang sie eingesetzt wird? Befindet sie sich in einem `deny`-Block, liefert sie den Grund für die Ablehnung an den Absender; in einem `warn`-Block erzeugt sie eine neue Header-Zeile. Diese Inkonsistenz ist ab Exim4, Version 4.6 abgeschafft: Dort gibt es jetzt die Konfigurationsoption `add_header`, mit der neue Header-Zeilen erzeugt werden können.

Wo wird nun das Makro gesetzt? Bei Debian-Systemen hängt es davon ab, ob Sie für die Konfiguration viele kleine Dateien oder eine große gewählt haben. Bei einer großen Konfigurationsdatei wird zusätzlich die Datei `/etc/exim4/exim4.conf.localmacros` ausgewertet, in der Sie einfach folgende Zeile eintragen können:

Listing 8.10: /etc/exim4/exim4.conf.localmacros

```
CHECK_RCPT_IP_DNSBLS = relays.ordb.org
```

Wir haben mit der **O**pen **R**elay **D**ata**B**ase einen der frei zugänglichen Server ausgewählt, die ein Verzeichnis der Spammer führen.

Bei vielen kleinen Dateien können Sie im Verzeichnis `/etc/exim4/conf.d/main/` eine Datei `000_localmacros` anlegen. Die drei Nullen sorgen dafür, dass die Makros zu Beginn eingelesen werden und damit allen folgenden Direktiven zur Auswertung zur Verfügung stehen.

8.8.3 Spam-Abwehr mit SpamAssassin

Der SpamAssassin bietet verschiedene Methoden, E-Mails inhaltlich auf Spam zu testen. Er bietet neben der Nutzung globaler Signaturen z. B. auch die Möglichkeit, sogenannte Bayes-Filter auf Nutzerbasis anzulernen. Wir werden nicht im Detail auf die vielfältigen Möglichkeiten und Funktionen des SpamAssassin eingehen, sondern uns auf seine Installation, Aktivierung und Nutzung mit Exim4 beschränken.

Der SpamAssassin besteht aus einem in Perl geschriebenen Dämon-Prozess, dem `spamd`, und einem in C geschriebenen Kommandozeilentool `spamc`, mit dem der Kontakt zum SpamAssassin-Dämon hergestellt werden kann. Diese Kombination bietet nach Ansicht der Autoren die beste Kombination aus Performance und Flexibilität.

Debian-Tipp

Um den SpamAssassin mit Exim4 nutzen zu können, muss das Paket `exim4-daemon-heavy` installiert sein, da Exim4 nur in dieser Variante mit der Möglichkeit kompiliert wurde, E-Mails an externe Validierer weiterzugeben. Um den SpamAssassin-Dämon zu aktivieren, muss zusätzlich in der Datei `/etc/default/spamassassin` die Option `ENABLED=1` gesetzt sein.

Der `spamd` wird i. d. R. über ein Init-Script gestartet. Für einen Test können Sie ihn mit `/etc/init.d/spamassassin start` per Hand starten; wollen Sie ihn regelmäßig einsetzen, sollten Sie ihn bereits beim Booten aktivieren. Bei allen aktuellen Distributionen geschieht dies automatisch.

Um die Lauffähigkeit zu testen, wird zum SpamAssassin eine Fake-Spam-Mail mitgeliefert. Sie finden diese im Verzeichnis `/usr/share/doc/spamassassin/examples`. Das Kommando `spamc` arbeitet auf relativ einfache Art und Weise: Es nimmt über die Standardeingabe eine E-Mail an, übermittelt diese an den `spamd` und liefert die von ihm bewertete Version auf der Standardausgabe zurück. Ein Test könnte so aussehen:

```
victor@hugo:~$ spamc < /usr/share/doc/spamassassin/examples/sample-spam.txt
Received: from localhost by hugo
        with SpamAssassin (version 3.1.1);
        Mon, dd Mmm 200n 17:50:40 +0200
From: Sender <sender@example.net>
To: Recipient <recipient@example.net>
Subject: Test spam mail (GTUBE)
Date: Wed, 23 Jul 2003 23:30:00 +0200
Message-Id: <GTUBE1.1010101@example.net>
X-Spam-Flag: YES
X-Spam-Checker-Version: SpamAssassin 3.1.1 (2006-03-10) on hugo
X-Spam-Level: **********************************************************
```

```
X-Spam-Status: Yes, score=1000.0 required=5.0 tests=GTUBE,NO_RECEIVED,
        NO_RELAYS autolearn=no version=3.1.1
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="----------=_44677C00.C351C953"

This is a multi-part message in MIME format.

------------=_44677C00.C351C953
Content-Type: text/plain; charset=iso-8859-1
Content-Disposition: inline
Content-Transfer-Encoding: 8bit

Software zur Erkennung von "Spam“ auf dem Rechner

    hugo

hat die eingegangene E-mail als mögliche "Spam"-Nachricht identifiziert.
[...]
victor@hugo:~$
```

Die ersten Zeilen der Ausgabe zeigen anhand der Header-Zeile `X-Spam-Flag: YES`, dass der `spamd` die E-Mail erkannt und passend bewertet hat. Jetzt müssen wir Exim4 dazu bewegen, eingehende E-Mails ebenfalls diese Schleife fliegen zu lassen:

```
[...]

spamd_address = 127.0.0.1 783

[...]
  # Add headers to a message if it is judged to be spam. Before enabling this,
  # you must install SpamAssassin. You also need to set the spamd_address
  # option in the main configuration.
  #
  # exim4-daemon-heavy must be used for this section to work.
  #
  warn
    spam = nobody
    message = X-Spam-Flag: YES\n\
              X-Spam_score: $spam_score\n\
              X-Spam_score_int: $spam_score_int\n\
              X-Spam_bar: $spam_bar\n\
              X-Spam_report: $spam_report
```

In der Konfigurationsdatei müssen dazu zwei Eintragungen vorgenommen werden, die i. d. R. aber schon enthalten sind und nur *auskommentiert* werden müssen. Es muss zum einen angegeben werden, auf welchem Port und auf welchem Server der

`spamd` auf Anfragen lauscht; und es muss natürlich konfiguriert werden, wie mit als Spam klassifizierten E-Mails umgegangen werden soll.

Exim4 kann im Übrigen mit der Ausgabe, die der SpamAssassin generiert, umgehen: die vom `spamd` generierten zusätzlichen Einträge werden von Exim4 abgeschnitten und ausgewertet. Damit die Information weitergegeben wird, müssen Sie in selbst definierten Header-Zeilen die ausgewerteten Informationen wieder in die E-Mail einfügen.

Debian-Tipp

Bei der Aufteilung in kleine Dateien müssen Sie einmal in `/etc/exim4/conf.d/main/02_exim4-config_options` die `spamd_address` und in `/etc/exim4/conf.d/acl/40_exim4-config_check_data` den `warn`-Block auskommentieren. Die Standardkonfiguration enthält die Zeile `X-Spam-Flag: YES` nicht, die wir ergänzt haben, um später im Client besser filtern zu können. Und vergessen Sie nach den Änderungen nicht, die Konfiguration mit `update-exim4.conf` zu aktualisieren!

Jetzt können Sie die Funktionsfähigkeit Ihres Spam-Filters testen: Werfen Sie noch einmal einen Blick in die Spam-Testmail:

Listing 8.11: /usr/share/doc/spamassassin/examples/sample-spam.txt

```
[...]
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.
```

Die letzten beiden Zeilen sind für uns interessant: Zum einen können Sie die Zeile mit den seltsamen Großbuchstaben-Kombinationen für einen Test nutzen, indem Sie sie kopieren und versenden; und Sie müssen es von einer *externen Mailadresse* aus tun. Warum dies? Bei einem Test auf Spam werden immer *alle* Teile der E-Mail ausgewertet, also auch die verschiedenen Header-Zeilen. Deuten diese auf eine »rein lokale Angelegenheit« hin, wird es Ihre Testmail nie auf ein passendes Spam-Level schaffen.

Achtung

Wundern Sie sich nicht, wenn die E-Mail ankommt und Sie erst einmal gar nichts erkennen können. Die Information, dass diese E-Mail als Spam klassifiziert wurde, ist nur in zusätzlichen Header-Zeilen untergebracht, und diese werden Ihnen vermutlich nicht angezeigt. Sie müssen in den Quelltext der E-Mail schauen, um das korrekte Funktionieren verifizieren zu können!

8.8.4 Attachments checken mit ClamAV

Der Virencheck funktioniert ähnlich wie bei der Spam-Erkennung mit einem Client-Server-Paar. Die eigentliche Aufgabe wird vom `clamd` erledigt; ein Test kann dann z. B. auf der Kommandozeile mit dem Kommando `clamdscan` aufgerufen werden. Zusätzlich existiert ein weiterer Dämon, der für die Aktualisierung der Virensignaturen sorgt. Dies ist i. d. R. schneller und verlässlicher als ein periodisch gestarteter Cron-Job.

Nach der Installation müssen Sie den Dämon starten – falls das nicht automatisch geschehen ist. Sie sollten ebenfalls den `freshclam`-Dämon starten, um die aktuellsten Signaturen herunterzuladen. Danach können Sie mit dem Kommando `clamdscan` Dateien auf Viren testen, z. B. eine der im Paket `clamav-testfiles` mitgelieferten:

```
victor@hugo:~$ clamdscan /usr/share/clamav-testfiles/clam.exe.bz2
/usr/share/clamav-testfiles/clam.exe.bz2: ClamAV-Test-File FOUND

----------- SCAN SUMMARY -----------
Infected files: 1
Time: 0.007 sec (0 m 0 s)
victor@hugo:~$
```

Achtung

Es existiert neben dem Kommando `clamdscan` noch ein Kommando `clamscan`. Dies ist jedoch die Standalone-Lösung, die ohne den `clamd`-Dämon arbeitet. Wenn dieses Kommando richtig arbeitet, ist das zwar schön, hilft uns aber nicht bei unserer Mail-Lösung weiter.

Jetzt können wir daran gehen, den automatischen Virenscan bei unserem Mailserver zu aktivieren: Wie beim Spam-Filter sind dafür im Prinzip zwei Eintragungen notwendig, die aber bereits vorbereitet sein sollten. Sie müssen also nur die passenden Einträge auskommentieren:

```
[...]

av_scanner = clamd:/tmp/clamd

[...]

  # Deny if the message contains malware. Before enabling this check, you
  # must install a virus scanner and set the av_scanner option in the
  # main configuration.
  #
```

```
# exim4-daemon-heavy must be used for this section to work.
#
deny
   malware = *
   message = This message was detected as possible malware ($malware_name).
```

Wir haben es hier mit einer relativ restriktiven Variante des Viren-Checks zu tun: soll eine E-Mail mit einem befallenen Anhang ausgeliefert werden, wird die Annahme mit dem obigen Kommentar abgelehnt. Dies ist aber i. d. R. kein Problem, weil der Absender in diesem Fall seine E-Mail zurückerhält und so nicht nur von der Ablehnung, sondern auch davon erfährt, dass er unter Viren leidet. Kommerzielle Lösungen bieten hier oft einen größeren Komfort, indem entweder der Virus aus den Anhängen entfernt oder die befallenen E-Mails gespeichert und bereitgehalten werden, um notfalls noch heruntergeladen werden zu können.

Debian-Tipp

Bei der Aufteilung in kleine Dateien müssen Sie wie bei der SpamAssassin-Konfiguration in `/etc/exim4/conf.d/main/02_exim4-config_options` die `av_scanner` und in `/etc/exim4/conf.d/acl/40_exim4-config_check_data` den `warn`-Block auskommentieren. Vorsicht bei der Angabe des Socket! Auf Debian-Systemen wird dieser in `/var/run/clamav/clamd.ctl` angelegt!

9 Federführend: Der Apache-Webserver

Der Apache-Webserver gehört mit zu den Erfolgsprojekten der Open-Source-Bewegung. Ohne ihn hätte es den Siegeszug von Linux im Server-Bereich wohl nicht gegeben. Zusammen mit der Scriptsprache PHP und der SQL-Datenbank MySQL ist die Kombination aus Linux und Apache über die Abkürzung LAMP ein Schlagwort für eine Webapplikationsumgebung geworden.

Mittlerweile gibt es den Apache-Server in einer Version 2, die wir in diesem Buch beschreiben wollen. Er ist modular aufgebaut. Das bedeutet, dass der Funktionsumfang über Module einfach erweitert werden kann; zum anderen wird er von den Distributionen oft in mehrere Pakete aufgeteilt. Die Paketübersicht zeigt Ihnen die Pakete, die für eine Basis-Installation notwendig sind. Außerdem haben wir die Pakete aufgeführt, die Sie für eine LAMP-Installation benötigen.

Basismodule einer Apache-Webserver-Installation	Debian	Fedora	SUSE
Grundfunktion Webserver mit allen grundlegenden Funktionen	apache2 apache2-common apache2-mpm-worker	httpd	apache2 apache2-worker
Kommandozeilentools für Apache Enthält u. a. die Tools `htpasswd` und `logrotate`	apache2-utils	in httpd enthalten	in apache2 enthalten
SSL-Modul Notwendig für den verschlüsselten Datenverkehr	apache2-common	mod_ssl	in apache2 enthalten
Apache-Dokumentation Sollte am Anfang immer mit installiert werden	apache2-doc	httpd-manual	apache2-doc
PHP Scriptsprache PHP in der Version 4, das P in LAMP	libapache2-mod-php4	php (Version 5)	libapache2-mod_php4
PHP-MySQL Für die Anbindung von PHP an MySQL-Datenbanken (MySQL-Pakete siehe Abschnitt 10.2)	php4-mysql	php-mysql	php4-mysql

Tabelle 9.1: Paketübersicht

9.1 Welche Grundfunktionen sind nach der Installation aktiviert?

Ein frisch installierter Webserver enthält eine Standardstartseite, die einem anzeigt, dass die Installation funktioniert hat. Diese Seite liegt typischerweise in einem Verzeichnis, das nur für die Ablage von Webseiten gedacht ist. Dieses Verzeichnis wird in den Konfigurationsdateien mit dem Namen `DocumentRoot` bezeichnet. In der Debian-Distribution liegt die `DocumentRoot` im Verzeichnis `/var/lib/www`. Sie müssen für einen ersten Test dort jedoch keine Datei ablegen; es reicht aus, den Webserver zu starten und die Startseite aufzurufen. Starten Sie einen Browser, und wählen Sie als Adresse `http://localhost`.

Meldet sich der Webbrowser mit einer Fehlermeldung, könnte es sein, dass der Server noch nicht läuft. Starten Sie ihn dann mit root-Berechtigung von der Kommandozeile aus über das Init-Skript. Der im Beispiel genutzte `su`-Befehl »switch user« bewahrt Sie davor, für diesen einen Befehl extra eine Konsole mit root-Berechtigung zu öffnen.

```
victor@hugo:~$ su -c '/etc/init.d/apache2 start'
Password:
Starting web server: apache2.
```

Abbildung 9.1: Apache-Startseite nach der Installation

```
victor@hugo:~$ firefox http://localhost/
victor@hugo:~$
```

Die jetzt angezeigte Seite (Abbildung 9.1) sehen Sie auch ab und zu einmal beim Surfen im Netz, wenn die Administratoren vergessen haben, die `DocumentRoot` auf das Verzeichnis mit den »richtigen Webseiten« umzusetzen – oder wenn die Seiten immer noch nicht fertig geworden sind. ;-)

9.1.1 Webbereiche für Normalnutzer

Um die Möglichkeiten zu testen, die der Apache-Server anbietet, ist in der Regel eine zweite Funktion aktiviert. Da der Bereich, auf den die `DocumentRoot` zeigt, für eine normale Nutzerkennung nicht schreibbar ist, können Sie Ihre ersten Gehversuche mit HTML-Seiten auch in Ihrem Home-Verzeichnis machen. Dort legen Sie ein Unterverzeichnis mit dem Namen `public_html` an. Jetzt können Sie dort HTML-Seiten speichern und über eine spezielle Adresse abrufen:

```
victor@hugo:~$ mkdir public_html
victor@hugo:~$ cd public_html
victor@hugo:~/public_html$ cat > index.html <<EOF
<html>
<head><title>Ein Test</title></head>
<body>
<h1>Ein Test</h1>
</body>
</html>
EOF
victor@hugo:~/public_html$ firefox http://localhost/~victor/
victor@hugo:~/public_html$
```

Wenn die Option `UserDir` angeschaltet ist, greifen Sie mit der speziellen Adresse `~username` auf die Dateien im Verzeichnis `/home/username/public_html` zu. Der Webserver ist außerdem so konfiguriert, dass er automatisch die Datei `index.html` ausgibt, wenn man bei der Adresse nur ein Verzeichnis angibt.

Haben Sie PHP direkt mitinstalliert? Dann wollen wir einmal schauen, ob das Modul auch »eingeschaltet« ist: Folgen Sie uns bitte noch einmal auf eine Kommandozeile mit root-Berechtigung:

```
hugo:~# cd /etc/apache2
hugo:/etc/apache2# ll mods-enabled
total 0
[...] cgid.conf -> /etc/apache2/mods-available/cgid.conf
[...] cgid.load -> /etc/apache2/mods-available/cgid.load
[...] userdir.conf -> /etc/apache2/mods-available/userdir.conf
[...] userdir.load -> /etc/apache2/mods-available/userdir.load
hugo:/etc/apache2#
```

Hier können Sie die Methode erkennen, mit der Module auf einem Debian-System ein- und ausgeschaltet werden. Im Verzeichnis `/etc/apache2/mods-available` werden die Konfigurationsdateien der Module abgelegt, die auf dem Server installiert sind. Durch einen symbolischen Link in das Verzeichnis `mods-enabled` werden die Module aktiviert. Damit man das nicht selbst machen muss, werden zum Apache-Server Tools mitgeliefert, die das Aktivieren erleichtern. In unserem obigen Beispiel ist PHP noch nicht eingeschaltet. Es bleibt Folgendes zu tun:

```
hugo:/etc/apache2# a2enmod php4
Module php4 installed; run /etc/init.d/apache2 force-reload to enable.
hugo:/etc/apache2# ll mods-enabled
[...] cgid.conf -> /etc/apache2/mods-available/cgid.conf
[...] cgid.load -> /etc/apache2/mods-available/cgid.load
[...] php4.conf -> /etc/apache2/mods-available/php4.conf
[...] php4.load -> /etc/apache2/mods-available/php4.load
[...] userdir.conf -> /etc/apache2/mods-available/userdir.conf
[...] userdir.load -> /etc/apache2/mods-available/userdir.load
hugo:/etc/apache2# /etc/init.d/apach2 force-reload
Forcing reload of web server: Apache2.
hugo:/etc/apache2#
```

SUSE-Tipp

SUSE bietet für die Konfiguration des Apache-Servers eine Datei `apache2` im Verzeichnis `/etc/sysconfig`. Darin wird eine Variable `APACHE_MODULES` deklariert, in der die zu startenden Module aufgelistet sind. In der Regel sind dort alle separat installierten Module auch direkt eingetragen, sodass Sie hier nicht mehr aktiv werden müssen. Sollten Sie in dieser Datei jedoch Änderungen vornehmen, vergessen Sie nicht, hinterher das Kommando `SUSEconfig` aufzurufen, das Ihre Änderungen in die Apache-Konfiguration übernimmt.

Fedora-Tipp

Am unbequemsten haben Sie es, wenn Sie Fedora einsetzen: Es existiert zwar ein Verzeichnis `/etc/httpd/conf.d`, in dem für die installierten Module Konfigurationsdateien gespeichert sind. Damit diese nicht geladen werden, müssen Sie die Dateien der unerwünschten Module so umbenennen, dass sie nicht auf `.conf` enden. Außerdem sind i. d. R. weitere Anpassungen in der zentralen Konfigurationsdatei `/etc/httpd/conf/httpd.conf` notwendig.

Das Kommando `a2enmod` (**a**pache**2** **en**able **mod**ules) legt die zur Aktivierung notwendigen Links an. Nach dem `force-reload` des Apache-Servers ist das PHP-Modul aktiviert, und wir können ausprobieren, ob es funktioniert: Dazu legen Sie eine einfache Textdatei mit dem Namen `index.php` im oben schon erwähnten Verzeichnis

`public_html` an, in der nur das PHP-Kommando `<? phpinfo() ?>` enthalten ist. Wenn Sie diese Datei mit einem Webbrowser aufrufen, wird Ihnen eine Webseite angezeigt, die Statusinformationen über das installierte PHP ausgibt.

Jetzt bleibt noch ein Test übrig: Kann der Apache-Server auch mit CGI-Scripts umgehen? Bei der Installation der Apache-Pakete wird das Verzeichnis `/usr/lib/cgi` (bei Fedora `/var/www/cgi-bin`) angelegt, das auch in der Konfiguration bereits voreingestellt und unter der Adresse `http://localhost/cgi"=bin/` zu erreichen ist. Dort können wir jetzt ein kleines Script ablegen und über den Webbrowser aufrufen:

```
hugo:~# cd /usr/lib/cgi-bin
hugo:/usr/lib/cgi-bin# cat > test.cgi <<EOF
#!/bin/sh
echo "Content-type: text/plain"
echo
echo Huhu!
EOF
hugo:/usr/lib/cgi-bin# chmod a+x test.cgi
hugo:/usr/lib/cgi-bin#
```

Nachdem wir ein einfaches CGI-Script angelegt und diesem auch Ausführungsrechte eingeräumt haben, sollte ein Aufruf von `firefox http://localhost/cgi-bin/test.cgi` Ihnen ein freundliches »Huhu!« liefern. Beachten Sie, dass wegen der Gefährdung durch schlecht oder falsch programmierte CGI-Scripts diese Verzeichnisse i. d. R. nur mit root-Berechtigung bestückt werden können.

Hinweis: Wenn's mal nicht so geklappt hat ...

Problem: Der Webserver beschwert sich beim Starten mit der Meldung »`Could not determine the server's fully qualified domain name, using 127.0.0.1 for ServerName`«.

Analyse: Darüber müssen Sie sich im Moment noch keine Gedanken machen. Der Apache-Server versucht anhand der IP-Nummer, unter der er läuft, einen passenden Namen zu suchen. Wenn er diesen nicht finden kann, beschwert er sich.

Lösung: Schauen Sie doch einmal, ob die Datei `/etc/hosts` existiert und dort eine Zeile mit dem Inhalt `127.0.0.1 localhost` enthalten ist. Wenn diese fehlt, tragen Sie sie einfach nach. Dann sollte diese Meldung nicht mehr auftreten.

Problem: Die Startseite ist auf Englisch, nicht auf Deutsch!

Analyse: Wenn man in das Startverzeichnis schaut, wird man feststellen, dass dort gar keine `index.html` existiert. Trotzdem liefert der Apache diese Seite aus – was sich auch einfach testen lässt, wenn man einmal statt `http://localhost/` die Adresse komplett mit dem Namen der HTML-Datei `http://localhost/index.html` angibt. Ursache für diesen geheimnisvollen Vorgang ist die Datei `index.html.var`, die vom Apache-Server ausgewertet wird. Dort ist für verschiedene Sprachen festgelegt, welche Version der `index.html` ausgeliefert werden sollen. Die Information darüber, welche Sprache nun gewünscht wird, nimmt der Apache aus dem Request: Das bedeutet in diesem Fall, dass der *Browser* eine andere Sprache als Deutsch angefordert hat!

Lösung: Versuchen Sie einmal, im Browser als Sprache Deutsch einzustellen. Das geht bei Firefox z. B. im Menü BEARBEITEN/EINSTELLUNGEN mit Hilfe des Buttons SPRACHEN bzw. auf Englisch LANGUAGES. Dort stellt man nicht die Sprache der Menüs des Browsers, sondern die Sprache der angeforderten Webseiten ein. Das hat bei normalem Surfen in den seltensten Fällen eine Auswirkung, weil i. d. R. die Unterstützung mehrsprachiger Webseiten nicht über diese Funktion realisiert ist. In unserem Fall sorgt es aber dafür, dass der Apache-Server eine passende Startseite herausgibt.

9.2 Grundkonfiguration für eine Website

Die Apache-Konfigurationsdateien waren ursprünglich dreigeteilt, und zwar

- in eine `httpd.conf`, die alle Direktiven für den Server enthielt,
- in eine `srm.conf`, in der alle Informationen über Verzeichnisse und Dateitypen enthalten waren, und

- in eine `access.conf`, in der Informationen über Zugriffsberechtigungen auf die verschiedenen Bereiche einer Site gespeichert waren.

Diese Dreifaltigkeit bestand, um Kompatibilität zu den ersten NCSA-Webservern zu gewährleisten. Der Apache-Server behandelt alle drei Dateien jedoch wie ein einziges, aus mehreren Teilen bestehendes Dokument. Daher ist heutzutage die Anzahl der Konfigurationsdateien nicht notwendigerweise vorgegeben. Sie werden in den verschiedenen Distributionen Varianten finden, die mit einer `httpd.conf` arbeiten; in manchen Fällen werden die anderen beiden Dateien als Dummies angelegt – mit dem Kommentar, dass sie eigentlich nicht mehr gebraucht werden.

Wir werden im Folgenden die aktuelle Konfiguration für eine Debian-Installation des Apache-Servers in der Version 2 beschreiben. Bei der Angabe der Konfigurationsdirektiven ist es unerheblich, in welcher Datei diese eingetragen sind; alleine die Reihenfolge der Auswertung ist bedeutsam. Wenn Sie also eine Installation betreuen, in der nur eine Datei oder eine andere Anzahl von Dateien genutzt werden, müssen Sie nur darauf achten, in welcher Reihenfolge die Optionen bei Ihnen abgearbeitet werden.

Die Debian-Paketbetreuer haben sich für den Apache eine stärker differenzierte Aufteilung der Konfigurationsdateien überlegt. Diese Aufteilung entstand aus den Bedürfnissen der Praxis von Paket- und Serverbetreuern, eine möglichst flexible Konfiguration zu erreichen und Teile der Standardkonfiguration von den lokal anzupassenden Teilen zu trennen. Damit ist es möglich, ein als separates Paket vorliegendes Modul zu installieren und zu aktivieren, ohne die mitgelieferten Konfigurationsschnipsel anpassen zu müssen, die i. d. R. standardisiert sind. Wir haben das im ersten Teil des Kapitels bereits demonstriert.

Lassen Sie uns einen näheren Blick auf das Verzeichnis `/etc/apache2` werfen, in dem die Konfiguration für den Apache-Webserver gespeichert ist:

```
hugo:/etc/apache2# ll
insgesamt 42K
-rw-r--r-- 1 root root  13K 2004-11-18 16:29 apache2.conf
drwxr-xr-x 2 root root 1,0K 2006-01-19 20:40 conf.d
-rw-r--r-- 1 root root  748 2005-02-09 22:29 envvars
-rw-r--r-- 1 root root  268 2004-11-18 16:29 httpd.conf
-rw-r--r-- 1 root root  13K 2004-11-18 16:29 magic
drwxr-xr-x 2 root root 3,0K 2006-02-02 19:01 mods-available
drwxr-xr-x 2 root root 1,0K 2005-06-08 21:01 mods-enabled
-rw-r--r-- 1 root root   10 2004-11-18 16:29 ports.conf
-rw-r--r-- 1 root root 2,3K 2004-11-18 16:29 README
drwxr-xr-x 2 root root 1,0K 2006-02-18 13:03 sites-available
drwxr-xr-x 2 root root 1,0K 2006-02-18 13:09 sites-enabled
hugo:/etc/apache2# egrep '^Include' apache2.conf
Include /etc/apache2/mods-enabled/*.load
Include /etc/apache2/mods-enabled/*.conf
Include /etc/apache2/httpd.conf
```

```
Include /etc/apache2/ports.conf
Include /etc/apache2/conf.d/[^.#]*
Include /etc/apache2/sites-enabled/[^.#]*
hugo:/etc/apache2#
```

Die Struktur der Konfigurationsdateien lässt sich auf einfache Weise demonstrieren, wenn man die zentrale Steuerdatei kennt: Die Datei `apache2.conf` enthält alle notwendigen Einträge, um die verschiedenen Dateien und Verzeichnisse einzubinden. Dies geschieht durch `Include`-Statements, die wir mit dem Kommando `egrep` herausgesucht haben. Das Ergebnis zeigt zugleich die Reihenfolge, in der die Dateien und Verzeichnisse abgearbeitet werden, da die Datei `apache2.conf` sequenziell durchlaufen wird.

Tipp

Wenn Ihre Distribution die Aufteilung in mehrere Dateien nicht direkt unterstützt, können Sie diese Aufteilung auch nachträglich selbst vornehmen. Dabei haben Sie es dann in der Hand, in wie viele kleine Dateien Sie Ihre Konfiguration aufteilen wollen! ;-)

mods-enabled/*.load: Für jedes nachladbare Apache-Modul müssen beim Laden im Prinzip zwei Aktionen durchgeführt werden: laden und konfigurieren. Die dafür notwendigen Schritte werden in separaten Konfigurationsdateien aufgeführt. In der Regel ist in den mit der Endung `.load` gekennzeichneten Dateien nur eine Zeile mit dem Ladebefehl enthalten. Aber auch hier gibt es Ausnahmen, z. B. in der Datei `mod_proxy.load`.

mods-enabled/*.conf: Das Laden eines Moduls reicht i. d. R. nicht aus, um das Modul nutzbar zu machen. Die für eine grundsätzliche Lauffähigkeit notwendigen Optionen sind in der Datei gleichen Namens mit der Endung `.conf` eingetragen.

mods-enabled und **mods-available**: Damit Sie sich diese Konfigurationsdateien nicht ausdenken müssen, werden bei der Installation Vorlagen im Verzeichnis `mods-available` gespeichert, die i. d. R. alle notwendigen Angaben für den Standardbetrieb enthalten. Da aber nur das Verzeichnis `mods-enabled` ausgewertet wird, müssen diese Dateien dort sichtbar gemacht werden. Das Kommando `a2enmod` tut dies mit Hilfe symbolischer Links; es wird also nur ein Verweis auf die Dateien im Verzeichnis `mods-available` angelegt. Mit dem Kommando `a2dismod` kann ein Modul wieder deaktiviert werden[1].

[1] Bei Debian-basierten Systemen werden bereits die Dateien im Verzeichnis `mods-available` als »Konfigurationsdateien« angesehen und auf Veränderungen kontrolliert; in einem solchen Fall werden diese Dateien bei einem Update nicht ohne Ihr Einverständnis überschrieben. Sie können auch die Dateien aus dem Verzeichnis `mods-available` in das Verzeichnis `mods-enabled` kopieren und die Kopien verändern. Das Kommando `a2dismod` wird diese Kopien jedoch löschen, wenn Sie das Modul damit deaktivieren!

```
hugo:~# ll /etc/apache2/mods-available/
                                                ...
-rw-r--r-- 1 root root 85 2005-10-24 05:52 /etc/apache2/mods-available/mime_magic.conf
-rw-r--r-- 1 root root 72 2004-11-18 16:29 /etc/apache2/mods-available/mime_magic.load
                                                ...
hugo:~# a2enmod mime_magic
Module mime_magic installed; run /etc/init.d/apache2 force-reload to enable.
hugo:~# ll /etc/apache2/mods-enabled/mime_magic.*
lrwxrwxrwx 1 root root 43 2006-02-27 17:18 /etc/apache2/mods-enabled/mime_magic.conf ->
/etc/apache2/mods-available/mime_magic.conf
lrwxrwxrwx 1 root root 43 2006-02-27 17:18 /etc/apache2/mods-enabled/mime_magic.load ->
/etc/apache2/mods-available/mime_magic.load
hugo:~# a2dismod mime_magic
Module mime_magic disabled; run /etc/init.d/apache2 force-reload to fully disable.
hugo:~# ll /etc/apache2/mods-enabled/
insgesamt ...
                                                ...
hugo:~#
```

Wie man in unserem Beispiel mit dem Modul `mime_magic` sehen kann, werden für beide Dateien symbolische Links angelegt. Zusätzlich werden wir noch freundlich darauf hingewiesen, was zu tun ist, um den Apache-Server zur Nutzung unserer Änderungen zu bewegen; wenn sich auf Ihrem Rechner das Kommando `apache2ctl` befindet, können Sie auch mit diesem den Server neu starten: `apache2ctl restart` sorgt für das gleiche Ergebnis.

httpd.conf: Diese Datei existiert bei Debian nur noch zu Kompatibilitätszwecken. Module von Drittherstellern, die ihre Software nicht direkt für das Debian-System vorbereiten, können so ohne Schwierigkeiten eingebunden werden. Die automatischen Installationsroutinen, die mit der Software mitgeliefert werden, legen i. d. R. Ergänzungen in dieser Datei ab.

```
hugo:/etc/apache2# cat httpd.conf
# This is here for backwards compatability reasons and to support
#  installing 3rd party modules directly via apxs2, rather than
#  through the /etc/apache2/mods-available,enabled mechanism.
#
#LoadModule mod_placeholder /usr/lib/apache2/modules/mod_placeholder.so
hugo:/etc/apache2#
```

ports.conf: In dieser Datei legen Sie fest, unter welchen IP-Adressen und auf welchen Ports sich Ihr Apache angesprochen fühlen soll. Hier ist ein großer Unterschied zur Version 1.x des Apache-Servers: Dort reichte die Angabe `*` als Schreiberleichterung, um alle Ports zu bestücken. Ab der Version 2 müssen hier jedoch eine oder mehrere Angaben gemacht werden.

```
hugo:/etc/apache2# cat ports.conf
Listen 80
hugo:/etc/apache2#
```

In der Standardkonfiguration lauscht der Apache-Server auf dem Standard-WWW-Port 80 auf allen hochgefahrenen Interfaces. Sie können zusätzliche Ports durch weitere `Listen`-Einträge ergänzen. Wenn Sie z. B. auch einen SSL-verschlüsselten Service über `https` anbieten wollen, müssen Sie noch `Listen 443` in der `ports.conf` eintragen.

`conf.d/[^.#]*`: In diesem Verzeichnis werden Konfigurationen abgelegt, global genutzt werden können. Ein Beispiel wäre das Debian-Paket `apache2-doc`: Nach der Installation finden Sie im Verzeichnis `conf.d` eine Datei, die die Apache-Dokumentation unter der Adresse `http://localhost/manual/` zur Verfügung stellt. Zur Sicherheit wird der Zugriff vorerst jedoch nur von der Adresse `127.0.0.1` erlaubt. Wenn Sie jetzt jedoch mehrere virtuelle Hosts für die IP-Adresse `127.0.0.1` anlegen, kann man für jeden dieser Hosts über die Adresse `http://.../manual` die Apache-Dokumentation abrufen.
Eine Besonderheit stellt der Suchausdruck `[^.#]*` dar: Er bedeutet, dass nur Konfigurationsdateien eingebunden werden, die keinen Punkt oder kein `#` als ersten Buchstaben haben. Wenn Sie also schnell mal eine in diesem Verzeichnis befindliche Konfiguration deaktivieren wollen, brauchen Sie nur den Dateinamen umzuändern:

```
hugo:/etc/apache2/conf.d# mv apache2-doc \#apache2-doc
hugo:/etc/apache2/conf.d# apache2ctl configtest
Syntax OK
hugo:/etc/apache2/conf.d# apache2ctl graceful
hugo:/etc/apache2/conf.d# lynx -dump http://localhost/manual

Not Found

The requested URL /manual was not found on this server.

_____________________________________________________________

Apache/2.0.55 (Debian) PHP/4.4.2-1 Server at localhost Port 80
hugo:/etc/apache2/conf.d#
```

`sites-enabled/[^.#]*` enthält die eigentlich wichtigen Konfigurationen. Hier sind in einzelnen Dateien die Einstellungen für Websites aus dem Verzeichnis `sites-available` verlinkt. In der Regel finden Sie dort nur die Datei `default`. Diese wird bei der Verlinkung speziell behandelt und bekommt noch ein paar Nullen vor den Dateinamen gesetzt, damit sie auch ganz bestimmt als Erstes geladen wird.

```
hugo:/etc/apache2# ll sites-available
insgesamt 2,0K
-rw-r--r-- 1 root root 1,2K 2006-01-16 11:15 default
hugo:/etc/apache2# ll sites-enabled
```

```
insgesamt 0
lrwxrwxrwx 1 root root 36 2006-02-27 19:04 000-default ->
/etc/apache2/sites-available/default
hugo:/etc/apache2#
```

Für den Suchausdruck `[^.#]*` gelten übrigens die gleichen Bedingungen wie beim Verzeichnis `conf.d`

Konfiguration der zentralen Webseiten

Jetzt haben Sie einen Überblick darüber, welche Konfigurationen in welchen Dateien und Verzeichnissen zu finden sind. Gehen wir aber zur Vertiefung noch einmal aus einer anderen Perspektive an das Problem »Konfiguration« heran:

- Wo müssen die Webseiten hin, die publiziert werden sollen?
- Wie kann man diese Webseiten abrufen?

Beide Fragen benötigen noch eine etwas ausführlichere Betrachtung.

9.2.1 Wo müssen die Webseiten hin, die publiziert werden sollen?

Das Startverzeichnis, in dem die Webseiten abgelegt sind, wird mit der Option `DocumentRoot` angegeben. Da dieses für verschiedene virtuelle Hosts unterschiedlich sein darf, taucht diese Option zum ersten Mal in der Datei `default` im Verzeichnis `/etc/apache2/sites-available` auf.

Listing 9.1: /etc/apache2/sites-available/default

```
DocumentRoot /var/www
<Directory />
   Options FollowSymLinks
   AllowOverride None
</Directory>
<Directory /var/www/>
   Options Indexes FollowSymLinks MultiViews
   AllowOverride None
   Order allow,deny
   Allow from all
</Directory>
```

Zusätzlich zur Angabe, wo die auszuliefernden Dateien gespeichert sind, finden sich in der Konfiguration noch Informationen darüber, wie dieser Bereich genutzt werden darf. Diese Informationen sind auf zwei Arten definiert:

- Mit `<Directory />` werden globale Einstellungen für den gesamten Dateibaum des Servers festgelegt. Die Direktive `Options` legt fest, Webseiten welcher Art wie ausgeliefert werden dürfen. Durch die Angabe `FollowSymLinks` wird z. B. festgelegt, dass bei der Angabe weiterer Konfigurationen für ein Verzeichnis gilt, dass vorhandene Seiten ausgeliefert werden und Symlinks nachgegangen wird. Symlinks

werden normalerweise nicht ausgeliefert, da sie in Bereiche des Dateisystems zeigen können, die eigentlich nicht öffentlich gemacht werden sollen und so ein Sicherheitsrisiko darstellen (siehe auch die Option `SymLinksIfOwnerMatch`). Das Fehlen von Optionen hat auch eine Auswirkung: Das Ausführen von CGI-Scripts z. B. ist dadurch generell abgeschaltet und muss für jedes Verzeichnis, in dem CGI-Scripts genutzt werden sollen, explizit eingeschaltet werden.

- Die Direktive `AllowOverride None` bestimmt, dass die Nutzung von `.htaccess`-Dateien (die wir in Abschnitt 9.5 erklären) generell abgeschaltet ist. So ist gewährleistet, dass niemand unbeobachtet die zentrale Konfiguration umgehen kann.
- Mit der Sektion `<Directory /var/www>` werden die relativ restriktiven Festlegungen der globalen Sektion etwas aufgeweicht. Als Optionen sind hier z. B. noch angegeben: `Indexes`, das die Nutzung von festgelegten Startseiten erlaubt, wenn in der URL nur das Verzeichnis angegeben wurde. Diese heißen i. d. R. `index.html`. Außerdem erlaubt diese Direktive, dass bei Fehlen einer solchen Startseite eine Übersicht über das angewählte Verzeichnis ausgegeben wird.
- `MultiViews` erlaubt die Nutzung verschiedener Darstellungsformen für eine angeforderte Seite. Die Mechanismen sind etwas zu kompliziert, um sie hier ausführlicher zu erläutern; kurz gesagt können mit Hilfe der Multiviews z. B. mehrsprachige Seiten realisiert werden. Die gewünschte Sprache wird dabei vom Browser übermittelt, und der Apache-Server liefert nach den Angaben des Browsers dann eine spezielle Datei aus, die nicht ganz anders heißt als die original angeforderte. Ein schönes Beispiel stellt die Standardseite dar, die bei frisch konfiguriertem Apache-Server ausgeliefert wird: Statt der angeforderten Datei `index.html`, die gar nicht existiert, orientiert sich der Server an den Einträgen in der Datei `index.html.var`. In dieser ist festgelegt, unter welcher Bedingung welche reale Seite auszuliefern ist. Möchte der Browser z. B. eine deutschsprachige Seite erhalten, wird die Datei `index.html.de` ausgeliefert.
- Die Optionen `Order allow,deny` und `Allow from` legen fest, welche IP-Adressen berechtigt sind, die Webseiten dieses Bereiches abzurufen. `Allow from all` legt hierbei fest, dass keine Einschränkungen bestehen. Die Direktive `Order` hat dabei nur insofern Einfluß auf die Konfiguration, indem sie festlegt, in welcher Reihenfolge `Allow` und `Deny`-Direktiven ausgewertet werden. Ein Szenario für eine gemischte Angabe von `Allow` und `Deny` Direktiven wäre das Zulassen bestimmter IP-Adressen bei Ablehnung aller anderen:

```
Order allow,deny
Allow from 192.168.1.
Allow from 192.168.2.
Deny from all
```

In diesem Beispiel dürften nur Rechner aus den Subnetzen 192.168.1.x und 192.168.2.x auf den Webserver zugreifen. Andersherum ginge es auch: Wenn Sie die IP-Adressen einiger »Bösewichte« kennen, denen Sie den Zugriff verwehren wollen, setzen Sie `Order deny,allow` und geben dann mit einer `Deny`-Direktive diese IP-Adressen an, gefolgt von `Allow from all`, damit allen anderen der Zugriff erlaubt ist.

Die Nutzung von CGI-Scripts wird durch eine besondere Direktive erlaubt: Mit `+ExecCGI` geben Sie an, dass in einem bestimmten Verzeichnisbereich ausführbare Dateien auch ausgeführt und nicht nur als solches ausgeliefert werden. Hier sehen Sie eine weitere Besonderheit: Findet der Apache-Server mehrere `Options`-Einträge, berücksichtigt er den letzten vollständig und ignoriert alle vorhergehenden. Wird den Angaben aber ein Plus oder ein Minus vorangestellt, werden die Optionen dem momentan gültigen Pool hinzugefügt respektive aus ihm abgezogen.

```
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
<Directory "/usr/lib/cgi-bin">
 AllowOverride None
 Options +ExecCGI -MultiViews
 Order allow,deny
 Allow from all
</Directory>
```

In dieser Konfiguration wird also die übergeordnete Konfiguration – Sie erinnern sich an die obige Sektion für das Verzeichnis `/`? – um die Optionen `ExecCGI` erweitert und `MultiViews` abgeschaltet.

Ohne die Direktive `ScriptAlias` hätten wir ein Problem: Das von uns angegebene Verzeichnis liegt nicht im Verzeichnisbaum, der als `DocumentRoot` angegeben wurde. Damit ein Zugriff möglich wird, müssen wir bestimmte URLs in den CGI-Bereich umlenken: Dies geschieht mit Hilfe der `ScriptAlias`-Direktive. Diese Direktive wird übrigens ausgewertet, bevor in der `DokumentRoot` nach einem passenden Verzeichnis bzw. einer Datei gesucht wird. Das heißt, dass eine Datei in einem Unterverzeichnis `/var/www/cgi-bin` nicht mehr erreichbar ist, weil der Apache-Server bei allen Anfragen, die in der URL mit `/cgi-bin` beginnen, nach einer Datei im Verzeichnis `/usr/lib/cgi-bin` sucht.

Für nicht-ausführbare Dateien – damit sind neben normalen HTML-Seiten z. B. auch PHP-Seiten gemeint – gibt es zur Direktive `ScriptAlias` das Gegenstück `Alias`.

```
Alias /doc/ "/usr/share/doc/"
<Directory "/usr/share/doc/">
 Options Indexes MultiViews FollowSymLinks
 AllowOverride None
 Order deny,allow
 Deny from all
 Allow from 127.0.0.0/255.0.0.0 ::1/128
</Directory>
```

Eventuell finden Sie obige Konfiguration noch in Ihrer Standardkonfiguration. Die `Alias`-Direktive lenkt nun alle Anfragen, die mit `/doc` beginnen, in das Verzeichnis `/usr/share/doc` um. Das ist notwendig, da sich das Verzeichnis `/usr/share/doc` ebenfalls nicht im Verzeichnisbaum des `DocumentRoot` befindet. Aber auch hier gilt, dass Dateien im Verzeichnis `/var/www/doc` nun nicht mehr erreicht werden können.

9.2.2 Wie kann man diese Webseiten abrufen?

In der Regel erwartet der Surfer eine Adresse in der Form `http://server.domain/verzeichnis/datei.html`. Um herauszufinden, auf welche Anforderungen unser Webserver reagiert, sind im Prinzip zwei Informationen notwendig: Für welche IP-Adresse fühlt er sich zuständig, und auf welchem Port lauscht er auf Anfragen? Bei namensbasierten virtuellen Servern ist hier noch eine dritte Information nötig; darauf gehen wir weiter unten noch ausführlicher ein. Für den Moment reicht uns ein Blick auf die `Listen`-Direktive, die sich in der Datei `ports.conf` befindet, um die gewünschten Informationen zu bekommen: `Listen 80` bedeutet hier, dass der Server auf allen mit IP-Adressen bestückten Netzwerkkarten auf dem Standard-WWW-Port lauscht. Dies sind i. d. R. die lokale Loopback-Schnittstelle mit der Adresse 127.0.0.1 und die IP-Adresse des Rechners, wenn Ihr Rechner in ein lokales Netzwerk eingebunden ist. Ein Zugriff über die Adresse `http://localhost` sollte also bereits ein Ergebnis bringen.

Nehmen wir einmal die Konfigurationen oben als Beispiel. Unser Server hat in unserem lokalen Netzwerk die IP-Adresse 192.168.1.10. Daher bekommen wir bei einem Aufruf von `http://192.168.1.10` die gleiche Seite wie beim Aufruf von `http://localhost`. Der Aufruf des Dokumentationsverzeichnisses, so wie es oben definiert ist, über die Adresse `http://localhost/doc` funktioniert nur mit `localhost`, aber nicht über die IP-Adresse `http://192.168.1.10/doc`. Dies geschieht aufgrund der Angaben in den `Allow` und `Deny`-Direktiven: Hier wird die »Absender«-Adresse ausgewertet, und die unterscheidet sich bei einem Zugriff über die lokale Loopback-Schnittstelle von derjenigen, die beim Zugriff auf die IP-Adresse 192.168.1.10 übermittelt wird.

Die wichtige Frage »Woher weiß ich denn, wer und wie viele Leute auf meine Seiten zugegriffen haben?« klären wir in Abschnitt 9.4.

9.3 Mehrere Websites mit einem Server verwalten

Namen sind zwar »Schall und Rauch«, aber eine schöne URL zieht man der schnöden Bezeichnung von `l736824.irgendein"=provider.com` vor, die Ihr root-Server oder der Router bei der Einwahl zugewiesen bekommt. Wenn Sie beruflich Webservices anbieten, werden schnell Wünsche nach unterschiedlichen Webadressen an Sie herangetragen. Was ist dafür notwendig? Brauchen wir dafür direkt einen neuen Server? Oder mehrere IP-Adressen? Oder geht das auch mit der einen IP-Adresse?

Nein, wir brauchen keinen neuen Server, und ja, es geht natürlich auch mit nur einer IP-Adresse[2]. Der Apache-Server besitzt die Möglichkeit, auf der Basis des Namens, unter dem er angesprochen wird, zwischen verschiedenen Bereichen zu differenzieren. Dies nennt man namensbasiertes, virtuelles Hosting. Es funktioniert

[2] Das soll aber nicht heißen, dass Sie nicht auch mit einem Server mehrere IP-Adressen bedienen könnten. Linux kann das, und der Apache-Server kann das natürlich auch.

mit allen neueren Browsern unter der Bedingung, dass der Browser den Servernamen mit übermittelt. Ältere Browser taten das nicht; wenn Sie also noch mit Mosaic oder Netscape 0.x arbeiten, könnten die folgenden Beispiele bei Ihnen nicht funktionieren.

Sie erinnern sich noch an die Datei `ports.conf`? Dort wurden die `Listen`-Direktiven eingetragen, die dem Apache-Server sagen, für welche Ports und IP-Adressen er zuständig ist. Im obigen Beispiel hatten wir die Variante mit der »Schreiberleichterung« genutzt und auf die Angabe der IP-Adressen verzichtet. Es kann aber helfen, hier die genutzten IP-Adressen mit anzugeben. Das erleichtert Ihnen hinterher die Übersicht darüber, wofür Ihr Apache-Server zuständig ist. Das ist insbesondere dann wichtig, wenn der Rechner mit mehreren IP-Adressen im Netz erscheint.

Eine »ausführliche« Konfiguration der Datei `ports.conf` sieht für die IP-Adresse 192.168.1.10 so aus:

Listing 9.2: ports.conf

```
Listen 192.168.1.10:80
NameVirtualHost 192.168.1.10:80
```

Neu dazugekommen ist die `NameVirtualHost`-Direktive. Sie sagt dem Apache-Server, dass auf der angegebenen IP/Port-Kombination Anfragen mit verschiedenen Servernamen kommen können. Damit sind die Grundlagen für die Einrichtung virtueller Hosts gegeben.

Tipp

Mehrere IP-Adressen können Sie einfach durch die Angabe mehrerer `Listen`-Direktiven nutzbar machen. Tragen Sie einfach jede IP-Adresse in eine neue Zeile ein. Sie können natürlich auch mehrere IP-Adressen als namensbasierte, virtuelle Hosts ansprechen. Ergänzen Sie dann für jede `Listen`-Direktive einen passenden `NameVirtualHost`-Eintrag.

Jetzt können wir dazu übergehen, im Verzeichnis `sites-available` neue Konfigurationsdateien für unsere virtuellen Hosts anzulegen. Am einfachsten ist es normalerweise, eine vorhandene Konfiguration zu kopieren und an den neuen Host anzupassen. Sie können z. B. die Datei `default` kopieren und ändern. Wir wollen aber einmal eine Datei von Grund auf mit allen notwendigen Einträgen aufbauen. Grundlage sei einmal das folgende Szenario: Wir wollen die Dokumentationen auf unserem Server unter der Adresse `docserver` im internen Netz zur Verfügung stellen. Es soll also hinterher unter der Adresse `http://docserver/` der Inhalt des Verzeichnisses `/usr/share/doc` zu erreichen sein.

VirtualHost: Konfigurationen für einen virtuellen Host werden mit dem Direktiven-Tag `VirtualHost` gekennzeichnet. Als Option muss dieser Direktive mitgegeben werden, unter welcher IP-Adresse und welchem Port der virtuelle Host arbeiten soll:

Listing 9.3: docserver

```
<VirtualHost 192.168.1.10:80>
</VirtualHost>
```

Diese Direktive umschließt unsere Konfiguration. Innerhalb der `VirtualHost`-Direktive werden über die Direktiven `ServerName` und `ServerAlias` die Namen des Servers angegeben, für die diese Konfiguration verantwortlich ist. Sie können jeweils nur eine `ServerName`-Direktive angeben, aber beliebig viele `ServerAlias`e. Zusätzlich sollten Sie für jeden virtuellen Host auch die Mailadresse des zuständigen Administrators angeben. Wenn Sie zusätzlich die Direktive `ServerSignature EMail` angeben, wird bei allen Verzeichnislisten und Fehlerseiten automatisch eine Fußzeile generiert, die unter anderem auch die unter `ServerAdmin` angegebene Mailadresse enthält.

Listing 9.4: docserver

```
<VirtualHost 192.168.1.10:80>
        ServerName docserver
        ServerAdmin victor@docserver
</VirtualHost>
```

Tipp

Sie können auch statt einer IP-Adresse in der `VirtualHost`-Direktive einen Hostnamen angeben. Das hat den Vorteil, dass sich der Apache-Server beschwert, wenn er den Namen nicht in eine IP-Adresse auflösen kann. In diesem Fall wird die komplette `VirtualHost`-Direktive ignoriert. Wenn der Name nur in `ServerName`- oder `ServerAlias`-Direktive angegeben wird, startet der Server-Prozess, ohne einen Hinweis auf eventuelle Probleme zu geben.

Im nächsten Schritt teilen wir dem Server mit, wo die Dateien sind, die er auf Anfrage auszuliefern hat. Die dafür notwendige Direktive `DocumentRoot` kennen Sie schon aus dem einleitenden Teil des Kapitels. Da wir alle Dateien aus dem Verzeichnis `/usr/share/doc` veröffentlichen wollen, ist dieses Verzeichnis auch unser root-Verzeichnis. Wir ergänzen die Angabe noch um eine `Directory`-Konfiguration, um festzulegen, wie die Daten in der `DocumentRoot` behandelt werden sollen und wer darauf zugreifen darf:

Listing 9.5: docserver

```
<VirtualHost 192.168.1.10:80>
        ServerName docserver
        ServerAdmin victor@docserver
 DocumentRoot /usr/share/doc
<Directory /usr/share/doc>
                Options Indexes FollowSymLinks
```

```
                AllowOverride None
                Order allow,deny
                allow from 192.168.1
deny from all
</Directory>

</VirtualHost>
```

Wir benötigen in unserem Szenario die Option `Indexes`, damit der Server aus unseren Verzeichnissen automatisch eine HTML-Seite generiert. Da viele Dokumentationen als separate Pakete installiert werden und Verlinkungen auf die »Originale« besitzen, machen wir uns mit `FollowSymlinks` das Leben etwas einfacher. Den IP-Bereich schränken wir jedoch auf das lokale Subnetz ein; es soll ja nicht jeder sehen, was wir alles für Pakete auf unserem Server installiert haben.

Jetzt sind wir so weit, einen ersten Test durchführen zu können: Als Voraussetzung muss nur gegeben sein, dass der Name `docserver` erreichbar ist. Wenn Sie keinen Zugriff auf den lokalen Nameserver haben, diesen nicht direkt verändern oder die Konfiguration nur testen wollen, reicht es aus, in der Datei `/etc/hosts` des Rechners, von dem aus Sie die Webseite testen wollen, IP-Adresse und Name zu ergänzen.

Jetzt muss die Konfigurationsdatei noch aktiviert werden: Wenn Sie die Konfigurationsdatei `docserver` genannt und im Verzeichnis `sites-available` abgelegt haben, können Sie dies mit dem Befehl `a2ensite` tun:

```
hugo:~# a2ensite docserver
Site docserver installed; run /etc/init.d/apache2 reload to enable.
hugo:~# apache2ctl configtest
Syntax OK
hugo:~# /etc/init.d/apache2 reload
hugo:~#
```

Damit wir keine Überraschungen erleben, haben wir vorher mit `apache2ctl` getestet, ob in der Konfiguration keine Fehler enthalten sind. Hier würde der Server auch meckern, wenn Sie bei der `VirtualHost`-Direktive einen Namen angegeben hätten, der nicht in eine IP-Adresse aufgelöst werden kann. Nach dem Neustart des Apache-Servers und allen Vorkehrungen bezüglich des neuen Namens sollte unter der Adresse `http://docserver/` jetzt eine Ausgabe des Verzeichnisses `/usr/share/doc` in Ihrem Browser erscheinen. Lassen Sie dem System etwas Zeit beim Aufbau der Seite, denn dieses Verzeichnis besitzt ziemlich viele Unterverzeichnisse.

9.4 Log-Dateien und Zugriffsstatistiken

Eine Direktive haben wir Ihnen bisher unterschlagen, die Ihnen bei der Durchsicht der Konfigurationsdateien aber bereits aufgefallen sein sollte: die verschiedenen Direktiven für die Erstellung von Log-Dateien. In der Regel wird jeder Zugriff vom

Apache-Server protokolliert. Dies dient nicht nur der Sicherheit; auch die Erstellung von Zugriffsstatistiken ist von der Existenz dieser Dateien abhängig. Diese Daten sind zu wichtig, um sie direkt bei der Entstehung auszuwerten und bei einem Systemabsturz dann Gefahr zu laufen, die aggregierten Daten zu verlieren. Daher wird die statistische Auswertung von Seitenzugriffen auch nicht vom Apache-Server selbst, sondern von separaten Tools wie Webalizer oder AWStats durchgeführt.

Was für Log-Dateien legt der Apache-Server während des Betriebs an? Prinzipiell würde eine einzige Datei ausreichen; es hat sich aber eingebürgert, Fehlermeldungen und Zugriffe getrennt zu protokollieren. Daher existieren auch zwei Direktiven, mit denen jeweils ein Zugriffslog und ein Fehlerlog angegeben werden können. Die `default`-Konfiguration enthält auf Debian-Systemen i. d. R. folgende Einträge bezüglich der Log-Dateien:

```
ErrorLog /var/log/apache2/error.log

# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
LogLevel warn

CustomLog /var/log/apache2/access.log combined
```

Das `ErrorLog`: Die Direktive erwartet eine Dateiangabe, in der die Log-Daten gespeichert werden sollen. Geben Sie am besten einen absoluten Pfad an; Pfade, die nicht mit einem »/« beginnen, werden relativ zu dem Verzeichnis ausgewertet, das über die Direktive `ServerRoot` in der Datei `apache2.conf` definiert ist. Steht am Anfang ein »/«, wird der Pfad so genommen, wie Sie ihn eingetragen haben.

Wichtig ist, dass das Verzeichnis existiert, in dem die Datei angelegt und gefüllt werden soll, der Server also dort unter der Kennung, unter der er läuft, Schreibrechte besitzt. Dies ist i. d. R. aber von den Distributionen bereits so eingerichtet und muss von Ihnen nur gesondert beachtet werden, wenn Sie Log-Dateien an anderen, unüblichen Stellen ablegen wollen.

Sie können die Ausführlichkeit, mit der der Server Angaben in das `Errorlog` schreibt, mit Hilfe der Direktive `LogLevel` steuern: Voreingestellter Wert bei Debian-Distributionen ist `warn`. Von den zur Verfügung stehenden Werten sind i. d. R. nicht alle relevant bzw. nur unter speziellen Bedingungen einzusetzen. Wichtig sind i. d. R. folgende Warn-Level, in abnehmender Ausführlichkeit und damit zu erwartender Größe der Log-Datei:

- `debug` gibt alle Meldungen aus, die in einem `ErrorLog` auftauchen können. Diese Option erzeugt große Log-Dateien und sollte von Ihnen nur genutzt werden, wenn Sie Probleme mit dem Server haben und testen wollen, woran es hapert.
- `warn` reduziert die Ausgabe von Meldungen auf Ereignisse, die den Status »Fehler« und »Warnung« haben. Selbst für Paranoiker reicht diese Einstellung im

laufenden Betrieb aus, um einen Eindruck von der Arbeitsweise des Servers zu bekommen.

- Wollen Sie nur Fehlermeldungen in Ihrem Error-Log haben, können Sie die Ausgabe durch die Angabe von `error` oder `crit` auf Fehler oder kritische Ereignisse beschränken. Beim Level `crit` kann es aber unter Umständen passieren, dass Ihnen wichtige Informationen durch die Lappen gehen: Es werden dann z. B. nicht gefundene Seiten nicht mehr aufgeführt.

In der Regel reicht es aber aus, den Server im normalen Betrieb mit `warn` oder `error` zu betreiben und bei Problemen auf `debug` zu schalten. Aber vergessen Sie nicht, im normalen Betrieb wieder zurückzuschalten, sonst läuft Ihnen Ihre Log-Partition voll!

Das `CustomLog` enthält die für eine statistische Auswertung relevanten Daten. Bei den Apache-Servern der Version 1.x hieß diese Datei noch `AccessLog`; über die Zeit hat es sich jedoch herausgestellt, dass eine standardisierte Ausgabe, wie sie von der AccessLog-Direktive vorgenommen wurde, nicht allen Anforderungen gerecht wurde. Daher gibt es jetzt das `CustomLog`, dessen Ausgabeform beliebig definiert werden kann. Damit Sie aber nicht jedes Mal überlegen müssen, wie Sie die Log-Zeilen aufbauen müssen, damit diese von der von Ihnen eingesetzten Auswertungssoftware auch verstanden wird, sind mit Hilfe der Direktive `LogFormat` unter den Schlüsselwörter `common` und `combined` die gebräuchlichsten Formate für Log-Dateien bereits definiert:

Listing 9.6: apache2.conf

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%h %l %u %t \"%r\" %>s %b \"%Refereri\" \"%User-Agenti\"" combined
```

Die Prozent-Operatoren stellen Platzhalter für spezifische Informationen dar, die beim Abruf einer Seite anfallen. Hier eine kurze Übersicht über die oben genutzten Operatoren:

- `%b` (**b**ytes): Anzahl der übertragenen Bytes
- `%h` (remote **h**ost): Der Name oder die IP-Adresse des Rechners, der die Seite abgerufen hat.
- `%i` (header lines): Bei einem Abruf einer Seite übermittelt der Browser eine Reihe von zusätzlichen Informationen in Form von Schlüssel/Wert-Paaren. Diese können abgerufen werden, indem man den Namen des Schlüssels in geschweiften Klammern dem `i` voranstellt. Beispiel: `%{Referer}i`
- `%l` (remote **l**ogname): Liefert die Kennung zurück, unter der der Abrufende auf seinem Rechner angemeldet ist. Das funktioniert nur, wenn die Direktive `IdentityCheck` aktiviert ist.
- `%r` (**r**equest): Der eigentliche Request, also z. B. der Abruf der Seite: "GET /index.html HTTP/1.0".
- `%s` (**s**tatus): Status des Abrufs: z. B. bei gelungener Auslieferung `200`.

- `%t` (**t**ime): Zeit des Abrufs.
- `%u` (remote **u**ser): Bei passwortgeschützten Seiten die Kennung, unter der sich der Abrufende angemeldet hat.

Eine Übersicht über alle nutzbaren Prozent-Operatoren und ihre Optionen finden Sie in der Dokumentation zum Modul `mod_log_config`.

Die oben als `common` definierte Log-Variante entspricht den ursprünglichen Ausgaben des `AccessLog` der Serverversion 1.x. Die vorgefertigte Regel `combined` beinhaltet den Referer, wenn er vom Browser übermittelt wird. Zusätzlich wird mit dem `User-Agent` das Programm gespeichert, das für den Abruf der Seiten genutzt wurde.

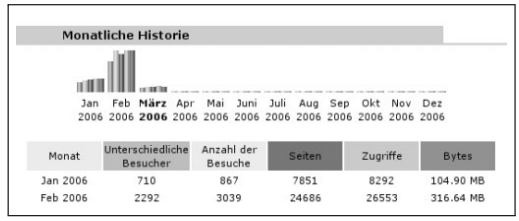
Exkurs

Die zunehmende Kommerzialisierung des Internet hat dazu geführt, dass die Anbieter von Inhalten mehr über die Nutzer ihrer Seiten erfahren wollten, als aus den Standard-Informationen zu entnehmen ist. Zwei wichtige Aspekte sollten zusätzlich gespeichert werden: *woher* die Nutzer kommen und *welches Programm* sie benutzen, wenn sie eine Webseite aufrufen. Das HTTP-Protokoll 1.0 (RFC 1945) von 1996 definiert bereits die Möglichkeit, den Request um zusätzliche Informationen zu erweitern, zu denen u. a. der *Referrer* gehört. Weil das Wort im offiziellen RFC aber falsch geschrieben wurde, muss und wird seitdem der *Referer* nur noch mit drei »r« geschrieben.

Es seien noch zwei Hinweise zum `combined`-Format erlaubt: Die Log-Dateien werden im Vergleich zum `common`-Format deutlich größer, da die zusätzlichen Informationen die Länge jedes Eintrags ungefähr verdoppeln. Außerdem müssen die dort übermittelten Informationen nicht stimmen: Bei einigen Browsern können Sie konfigurieren, als welcher Browser-Typ sie sich ausgeben sollen – ein Überbleibsel aus den Zeiten der Browser-Kriege, in denen manche Webseiten sich weigerten, mit bestimmten Browsern zusammenzuarbeiten und dazu diese Information auswerteten. Außerdem werden die Referer-Informationen von Akteuren im Internet genutzt, um den Eindruck zu vermitteln, auf ihren Webseiten wären Links auf Ihre Webseiten enthalten. Wenn Sie jetzt nachschauen wollen, was und wie derjenige Ihre Seiten wohl verlinkt hat, werden Sie zum einen nicht fündig, erhöhen andererseits aber die Abrufzahlen des »Referer-Spammers«. Daher sollten Sie die Informationen, die Sie mit Hilfe des `combined`-Logs gewinnen, nicht überbewerten und sich vorher überlegen, ob Ihnen eine statistische Auswertung dieser Daten überhaupt sinnvoll erscheint.

9.4.1 Auswertungsprogramme für Zugriffslogs

Es gibt eine ganze Reihe von Programmen, die versuchen, lange Listen von Seitenabrufen in eine nette grafische Form zu bringen. Da alle Tools i. d. R. den gleichen Funktionsumfang besitzen, bleibt der aus der statistischen Verdichtung der Rohdaten entstehende Informationsgehalt der Gleiche; Sie sollten daher anhand von Beispielausgaben der einzelnen Tools für sich selbst entscheiden, welches Sie ein-



Monatliche Historie

Jan 2006 Feb 2006 **März 2006** Apr 2006 Mai 2006 Juni 2006 Juli 2006 Aug 2006 Sep 2006 Okt 2006 Nov 2006 Dez 2006

Monat	Unterschiedliche Besucher	Anzahl der Besuche	Seiten	Zugriffe	Bytes
Jan 2006	710	867	7851	8292	104.90 MB
Feb 2006	2292	3039	24686	26553	316.64 MB

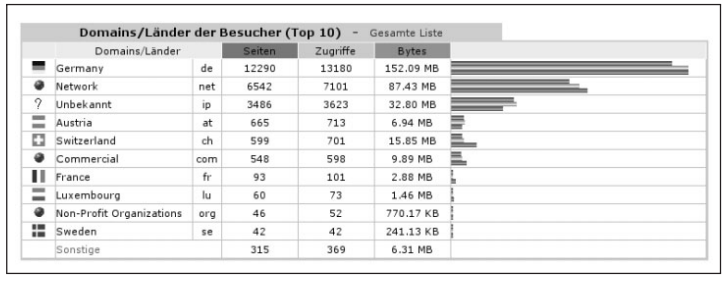
Abbildung 9.2: awstats: Monatsübersicht

setzen wollen. Da die Tools die bereits ausgewerteten Daten aber i. d. R. aggregieren und in eigenen Formaten speichern, ist ein nachträglicher Wechsel ohne die Original-Rohdaten kompliziert und nicht immer möglich. Solange Sie die Log-Dateien jedoch archivieren, können Sie später ohne Probleme zu einem anderen Tool wechseln.

Wir wollen Ihnen hier kurz das Tool awstats vorstellen, das wir neben dem Tool webalizer bei uns im Einsatz haben. Es besteht aus einem Perl-Script plus ein paar Konfigurationsdateien, ist relativ einfach gehalten und legt die gesammelten Informationen in Text-Dateien ab, die im Falle eines Falles auch von einem »menschlichen Leser« ausgewertet werden können. Die Reports werden durch das gleiche Script dynamisch aus den gespeicherten Aggregat-Daten erstellt und sind daher beim jedem Aufruf der Auswertungsseite aktuell.

Ein Report beginnt i. d. R. mit einer Monatszusammenfassung, die von einer Tagesübersicht ergänzt wird. Darunter finden sich dann genauere Aufschlüsselungen zu Tageszugriffen oder zu IP-Adressen, von denen aus die Seiten abgerufen wurden. Außerdem finden Sie eine Statistik darüber, welche Seiten am häufigsten besucht wurden, erfahren aber auch, welche Seiten insgesamt wie häufig abgerufen wurden.

Was ist für die Einrichtung notwendig? Wenn Sie awstats aus den Paketen der Distribution installieren, sollten alle notwendigen Konfigurationen bereits angelegt sein. Dazu gehört bei Debian-Systemen eine Konfigurationsdatei im Verzeichnis `/etc/apache2/conf.d`, die Direktiven für die Verzeichnisse mit Icons und Daten enthält. Dort wird auch ein `ScriptAlias` mit Namen `/awstats` definiert, das auf das Verzeichnis `/usr/lib/cgi-bin` zeigt, in dem das Haupt-Perl-Script installiert ist.



Domains/Länder der Besucher (Top 10) - Gesamte Liste

Domains/Länder		Seiten	Zugriffe	Bytes
Germany	de	12290	13180	152.09 MB
Network	net	6542	7101	87.43 MB
Unbekannt	ip	3486	3623	32.80 MB
Austria	at	665	713	6.94 MB
Switzerland	ch	599	701	15.85 MB
Commercial	com	548	598	9.89 MB
France	fr	93	101	2.88 MB
Luxembourg	lu	60	73	1.46 MB
Non-Profit Organizations	org	46	52	770.17 KB
Sweden	se	42	42	241.13 KB
Sonstige		315	369	6.31 MB

Abbildung 9.3: awstats: Domain-Übersicht

Angepasst werden muss von Ihnen nur die zentrale Konfigurationsdatei `/etc/awstats/awstats.conf`. Auf Debian-Systemen enthält diese Datei ein `Include`-Statement auf eine Datei mit Namen `awstats.conf.local`. Sie können daher auch alle Anpassungen in dieser Datei separat speichern. Wenn Sie virtuelle Hosts nutzen, können Sie die für die Analyse wichtigen Optionen auch in Dateien der Form `awstats.www.meine.domain.conf` ablegen. Um eine separate Log-Datei für unser `docserver`-Beispiel anzulegen, müssen Sie die Konfigurationsdatei `awstats.docserver.conf` anlegen und folgendermaßen bestücken:

```
LogFile="/var/log/apache2/docserver-access.log"
LogFormat=1
SiteDomain="docserver"
HostAliases="localhost 127.0.0.1 192.168.1.10"
SkipHosts=""
DirData="/var/lib/awstats"
```

Tipp

Das Paket `awstats` enthält ein Script `awstats_configure.pl`, das interaktiv für Sie eine Konfigurationsdatei generiert. Auf Debian-Systemen finden Sie es im Verzeichnis `/usr/share/doc/awstats/examples`. Damit können Sie eine Basis-Datei erzeugen, die i. d. R. alle wichtigen Einstellungen bereits enthält und nur geringfügig von Ihnen angepasst werden muss.

Wichtigster Eintrag ist der Verweis auf die Log-Datei, die ausgewertet werden soll. An dieser Stelle haben wir Ihnen noch etwas unterschlagen. Wenn Sie sich an die Konfiguration oben erinnern, haben wir dem virtuellen Host keine Log-Direktiven mitgegeben. Der Apache-Server ist in diesem Zusammenhang konsequent: Ohne `CustomLog` oder `ErrorLog` werden *keine* Zugriffe oder Fehler protokolliert. Ergänzen wir also unsere VirtualHost-Konfiguration um die notwendigen Einträge:

Listing 9.7: docserver

```
<VirtualHost 192.168.1.10:80>
        ServerName docserver
        ServerAdmin victor@docserver
DocumentRoot /usr/share/doc
<Directory /usr/share/doc>
                Options Indexes FollowSymLinks
                AllowOverride None
                Order allow,deny
                allow from 192.168.1
deny from all
</Directory>
```

```
        ErrorLog /var/log/apache2/docserver-error.log
        LogLevel warn
        CustomLog /var/log/apache2/docserver-access.log combined

</VirtualHost>
```

Jetzt sollten Sie den Server mit `apache2ctl graceful` neu starten.

Kommen wir zurück zur awstats-Konfiguration: Die Direktive `LogFormat` legt fest, in welcher Form der Apache-Server seine Log-Daten speichert. Die `1` steht für das von uns ausgewählte `combined`-Format. Sie können awstats aber auch durch Prozent-Operatoren mitteilen, wie Sie das Logformat für den Apache-Server festgelegt haben. Leider sind diese Operatoren anders aufgebaut, als in den Apache-LogFormat-Direktiven. Eine vollständig Dokumentation finden Sie in der Datei `/etc/awstats/awstats.conf` oberhalb der `LogFormat`-Direktive.

`SiteDomain` definiert, für welche Domain diese Konfiguration zuständig ist. Diesen Eintrag müssen Sie auf jeden Fall konfigurieren, wenn Sie alle Ihre virtuellen Server in eine Log-Datei schreiben lassen. Nur dann ist awstats in der Lage, die Log-Einträge der verschiedenen virtuellen Hosts voneinander zu unterscheiden. Sie müssen natürlich auch noch dafür sorgen, dass die virtuellen Hosts in der Log-Datei verewigen, vom welchem der Eintrag nun stammt. Dazu existiert die Option `%virtualname` für die Anpassung des Apache-Logs. Wir würden Ihnen aber immer dazu raten, für jeden virtuellen Host separate Log-Dateien anzulegen. In einem solchen Fall hat `SiteDomain` nur die Funktion, korrekte URLs auf den Ausgabeseiten von awstats zu generieren.

Die Direktiven `HostAliases` und `SkipHosts` dienen dazu, Zugriffe von bestimmten Servern aus der Berechnung auszuschließen. Die Idee dahinter ist, dass Zugriffe vom selben Server oder von den in `SkipHosts` explizit angegebenen Servern administrativer Art sind und die Statistik über die eigentlichen Zugriffe verfälschen würden. Wenn Sie z. B. ausschließen wollen, dass Zugriffe von innerhalb der Domain mitgezählt werden, weil sich dahinter nur Zugriffe verbergen, mit denen die Webseiten gepflegt oder z. B. die statistischen Auswertungen angesehen werden, sollten Sie die möglichen IP-Adressen durch Leerzeichen getrennt in `SkipHosts` aufführen. Außerdem ist die Angabe von regulären Ausdrücken mit Hilfe von `REGEX[]` möglich.

Damit haben wir unser Möglichstes getan, um unsere Statistiken selbst zu fälschen und können awstats jetzt überreden, die Auswertungen vorzunehmen. Dazu rufen wir das zentrale awstats-Perl-Script auf und geben ihm die Konfiguration mit, die wir gerade erstellt haben:

```
hugo:~# /usr/lib/cgi-bin/awstats.pl -config=docserver
Update for config "/etc/awstats/awstats.docserver.conf"
With data in log file "/var/log/apache2/docserver-access.log"...
Phase 1 : First bypass old records, searching new record...
Searching new records from beginning of log file...
```

```
Phase 2 : Now process new records (Flush history on disk after 20000 hosts)...
Jumped lines in file: 0
Parsed lines in file: 219
 Found 0 dropped records,
 Found 0 corrupted records,
 Found 0 old records,
 Found 219 new qualified records.
hugo:~# /usr/lib/cgi-bin/awstats.pl -config=docserver
Update for config "/etc/awstats/awstats.docserver.conf"
With data in log file "/var/log/apache2/docserver-access.log"...
Phase 1 : First bypass old records, searching new record...
Direct access after last parsed record (after line 219)
Jumped lines in file: 219
 Found 219 already parsed records.
Parsed lines in file: 0
 Found 0 dropped records,
 Found 0 corrupted records,
 Found 0 old records,
 Found 0 new qualified records.
hugo:~#
```

An diesem Beispiel lassen sich zwei Dinge erkennen: awstats benötigt, um die passende Konfigurationsdatei herauszufinden, nur den »Mittelteil« – vorausgesetzt, wir haben uns an die Konvention gehalten und die Konfigurationsdatei richtig benannt. Am zweiten Aufruf sehen Sie, dass awstats sich den aktuellen Stand merkt und alle Einträge überspringt, die älter sind bzw. bereits ausgewertet wurden.

Wo landet die interne Datenbank? Wenn Sie wie in unserem Beispiel die Option `DirData` angegeben haben, finden Sie in diesem Verzeichnis eine Datei der Form `awstats<Datum>.docserver.txt`. Diese Datei enthält die kumulierten Daten der letzten Auswertungen und die für awstats wichtigen Informationen, wo und wie es bei den nächsten Aufrufen weiter auswerten soll. Das bedeutet für uns:

- Von dieser Datei können wir Backups machen. Sie reicht aus, um die Auswertungen zu erzeugen.
- Wenn wir einen erneuten Durchlauf mit den originalen Log-Dateien machen wollen, reicht es aus, diese Datei zu löschen und awstats die (alten) Log-Dateien noch einmal vorzusetzen.

Man könnte z. B. die alten Dateien zu einer zusammenfügen, die interne Datenbankdatei löschen und awstats eine erneute Analyse durchführen lassen:

```
hugo:~# ls docserver*
docserver.2003.gz
docserver.2004.gz
docserver.2005.gz
hugo:~# zcat docserver.*.gz > /var/tmp/docserver.old.log
```

```
hugo:~# rm /var/lib/awstats/awstats*.docserver.txt
hugo:~# /usr/lib/cgi-bin/awstats.pl -config=docserver -LogFile=/var/tmp/docserver.old.log
                                                    ...
hugo:~#
```

Durch die Option `LogFile` überstimmen wir für diesen einen Aufruf die `LogFile`-Direktive unserer docserver-Konfiguration.

Für die Generierung der HTML-Seiten benötigt awstats die Information, unter welcher Adresse das CGI-Script `awstats.pl` aufgerufen wird und wie es an seine Icons herankommt. Diese Daten sind in einem Konfigurationsschnipsel gespeichert, das auf Debian-Systemen im Verzeichnis `/etc/apache2/conf.d/` installiert ist und `awstats.conf` heißt.

Listing 9.8: /etc/apache2/conf.d/awstats

```
                                                    ...
# This provides worldwide access to everything below the directory
# Security concerns: none known
<Directory /usr/share/awstats/icon>
        Options None
        AllowOverride None
        Order allow,deny
        Allow from all
</Directory>

# This provides worldwide access to everything in the directory
# Security concerns: none known
Alias /awstats-icon/ /usr/share/awstats/icon/

# This (hopefully) enables _all_ CGI scripts in the default directory
# Security concerns: Are you sure _all_ CGI scripts are safe?
ScriptAlias /awstats/ /usr/lib/cgi-bin/
                                                    ...
```

Dort finden Sie einen `ScriptAlias`-Eintrag, der auf `/usr/lib/cgi-bin` weist und einen `Alias`-Eintrag, der das Verzeichnis mit den awstats-Icons `/usr/share/awstats/icon` unter `/awstats"=icon` zur Verfügung stellt. Diese Informationen werden in der awstats-Konfiguration unter `DirCgi` und `DirIcons` definiert.

```
LogFile="/var/log/apache2/docserver-access.log"
LogFormat=1
SiteDomain="docserver"
HostAliases="localhost 127.0.0.1 192.168.1.10"
SkipHosts=""

DirCgi="/awstats"
DirIcons="/awstats-icon"
```

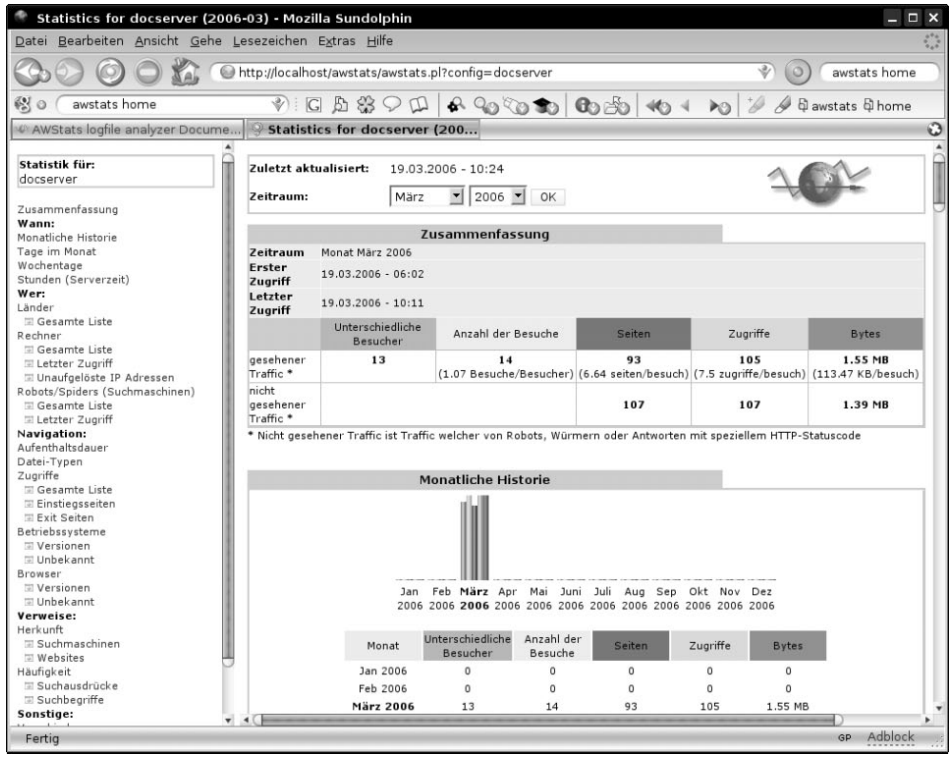


Abbildung 9.4: awstats: Statistikseite

Jetzt können Sie unter der Adresse `http://localhost/awstats/awstats.pl?config=docserver` auf die Statistikseite zugreifen (Abbildung 9.4).

Regelmäßige Updates der internen Datenbank: Damit Sie auf der Webseite immer den aktuellen Stand ablesen können, ist es notwendig, dass awstats die interne Datenbank regelmäßig auffrischt. Dazu wird bei der Installation im Verzeichnis `/etc/cron.d` ein cron-Eintrag für die Standardkonfiguration abgelegt:

```
0,10,20,30,40,50 * * * * www-data [ -x /usr/lib/cgi-bin/awstats.pl -a
-f /etc/awstats/awstats.conf -a -r /var/log/apache2/access.log ] &&
/usr/lib/cgi-bin/awstats.pl -config=awstats -update >/dev/null
```

Dieser Eintrag sieht ziemlich wild aus und muss in einer Zeile stehen. Wir wollen hier nicht auf die Details eines cron-Eintrags eingehen; aber ein paar Kommentare zu der Funktionsweise dieses Aufrufs seien erlaubt:

- Die Angaben in den eckigen Klammern prüfen, ob alle notwendigen Dateien vorhanden sind, um eine erfolgreiche Auswertung durchzuführen. Es wird abgeprüft, ob das Perl-Script `awstats.pl` ausführbar ist und ob die Konfigurations- und die Log-Dateien existieren. Erst dann wird awstats gestartet.

- Das Script `awstats.pl` wird auf Debian-Systemen mit der Kennung `www-data` gestartet. Diese Kennung entspricht derjenigen, unter der der Apache-Server läuft und ermöglicht awstats damit Zugriff auf alle wichtigen Dateien, ohne mit root-Berechtigung laufen zu müssen.

Um den Cron-Job an unsere `docserver`-Konfiguration anzupassen, kopieren wir die Datei (z. B. unter dem Namen `awstats-docserver`) und ändern Teile der Konfiguration wie folgt:

```
0,10,20,30,40,50 * * * * www-data [ -x /usr/lib/cgi-bin/awstats.pl -a
-f /etc/awstats/awstats.docserver.conf -a -r /var/log/apache2/docserver-access.log ] &&
/usr/lib/cgi-bin/awstats.pl -config=docserver -update >/dev/null
```

Jetzt wird alle zehn Minuten ein Update der statistischen Auswertung durchgeführt. Wenn Sie Ihren Server nicht unnötig mit Auswertungen belasten wollen und Ihnen eine halbstündige Aktualisierung ausreicht, können Sie die Update-Frequenz auch heruntersetzen und `0,10,20,30,40,50` z. B. durch `0,30` ersetzen.

Und noch ein paar Sätze zur Sicherheit. Beginnen wir mit der Dateisystemebene: Das erste Problem, das i. d. R. auftritt, lässt sich darauf zurückführen, dass Sie zum Testen oder auch aus anderen Gründen die interne Datenbank unter der root-Kennung erzeugt haben. Wenn die Option `SaveDatabaseFilesWithPermissionsForEveryone=0` in Ihrer `docserver`-Konfiguration enthalten ist, wird der Cron-Job, wenn er wie oben unter der Kennung `www-data` gestartet wird, in dieser Datei keine Änderungen vornehmen können. Daher sollten Sie den Besitzer der internen Datenbank auf `www-data` setzen und per Hand gestartete Updates immer mit `su` durchführen.

Ist die Option `SaveDatabaseFilesWithPermissionsForEveryone` in Ihrer Konfigurationsdatei nicht enthalten und haben Sie keine Probleme, dann sollten Sie sich die Rechte der internen Datenbank einmal genauer ansehen. Unter Umständen nimmt awstats als Default an, dass die Datenbank-Datei mit Lese- und Schreibrechten für alle Systemnutzer gespeichert werden soll. In diesem Fall sollten Sie die Konfigurationsoption mit dem Wert 0 nachtragen und die Rechte »zu Fuß« mit `chmod go-w` ändern, damit nur der Nutzer `www-data` diese Datei verändern kann.

Wenn Sie weitere CGI-Scripts im Verzeichnis `/usr/lib/cgi-bin` abgelegt haben, die ohne Einschränkung genutzt werden sollen, können Sie den Zugriff auf awstats nur schlecht einschränken. In einem solchen Fall wäre es besser, das Perl-Script `awstats.pl` in einem anderen Verzeichnis unterzubringen, dieses per `ScriptAlias` unter dem Namen `/awstats` zur Verfügung zu stellen und für dieses Verzeichnis spezielle Zugriffseinschränkungen zu definieren. Dies können einerseits Beschränkungen auf bestimmte IP-Adressen sein, andererseits aber auch eine Zugangskontrolle mit Hilfe von Kennung und Passwort, wie wir im nächsten Kapitel erklären werden.

Weitere Funktionen von awstats finden Sie in der im Paket enthaltenen Dokumentation oder online. Dort ist u. a. beschrieben, wie man statische HTML-Seiten als Ausgabe erzeugen kann; diese haben den Vorteil, dass man sie an anderer Stelle speichern und abrufen kann, ohne dass dort ein vollständiges awstats installiert sein muss.

Weitere Literatur

- awstats-Homepage: `http://awstats.sourceforge.net/`

9.5 Geschützte Bereiche einrichten

Bisher haben wir unsere Seiten nur durch Regeln geschützt, die sich auf IP-Adressen beziehen. Mit der Auswertung unserer Log-Dateien haben wir aber ein erstes Szenario, in dem wir einen personenbezogenen Zugriffsschutz in Form von Kennung/Passwort-Paaren realisieren wollen. Was ist dazu notwendig?

- Eine Datei mit Kennungen und Passwörtern
- Ein Bereich, der geschützt werden soll
- Eine Direktive, die die Passwortdatei einbindet

Beginnen wir mit dem Anlegen einer Passwortdatei. Die vom Apache-Modul `mod_auth_basic` (das in den Vorgängerversionen nur `mod_auth` hieß) benötigte Datei für die Autorisierung ähnelt den Dateien `/etc/passwd` bzw. `/etc/shadow`, in denen die Systemkennungen und Passwörter gespeichert werden. Leider ähnelt sie diesen nur, was im Umkehrschluss heißt, dass wir den Apache-Server nicht direkt auf eine dieser Dateien loslassen können.

Wir müssen also eine eigene Datei für unsere Autorisierung anlegen und pflegen. Dazu steht uns das Kommando `htpasswd` zur Verfügung, mit dem wir Kennungen anlegen, ändern und löschen können:

```
hugo:/usr/lib/cgi-bin# htpasswd -mc .htpasswd victor
New password:
Re-type new password:
Adding password for user victor
hugo:/usr/lib/cgi-bin# ll .htpasswd
-rw-r--r-- 1 root root 45 2006-03-19 16:53 .htpasswd
hugo:/usr/lib/cgi-bin# cat .htpasswd
victor:$apr1$28lyB...$bxi694.agqjekOXT1QoC.0
hugo:/usr/lib/cgi-bin#
```

Die Optionen bei einem Aufruf von `htpasswd` sehen so aus: Zuerst legen Sie mit (oder ohne) eine Option die Art der Verschlüsselung des Passworts fest, danach geben Sie die zu bearbeitende Passwortdatei an; am Ende steht die Kennung, die verändert werden soll. Existiert die Kennung noch nicht in der ausgewählten Datei, wird sie neu angelegt, ansonsten wird das Passwort geändert.

Die wichtigsten Optionen im Überblick:

- `-c` sagt dem Kommando `htpasswd`, dass die angegebene Datei neu angelegt werden soll.
- `-m` legt fest, dass das Passwort md5-verschlüsselt gespeichert werden soll. Zusätzlich stehen noch folgende Verschlüsselungsmodi zur Verfügung: Als Vor-

einstellung gilt die Verschlüsselung mit Crypt, mit `-s` können Sie das Passwort SHA-kodiert speichern, und `-p` speichert das Passwort im Klartext.

- `-D` löscht die angegebene Kennung.

Passwortdateien können Analysen unterzogen werden, die in der Lage sind, schwache Passwörter zu knacken. Daher sollten Dateien, die Passwörter enthalten, i. d. R. nur von denjenigen gelesen werden können, die den Zugriff unbedingt brauchen. In unserem Fall bedeutet das, dass eigentlich nur der Apache-Server darauf zugreifen können sollte. Mit unserer Passwortdatei haben wir daher noch zwei Probleme:

1. Sie ist im Moment noch von allen Systemnutzern lesbar. Das lässt sich schnell ändern. Jedoch gehört sie noch der root-Kennung und könnte nach ändern der Rechte vom Apache-Prozess nicht mehr gelesen werden. Daher müssen wir der Datei Besitzer und Gruppe des Apache-Prozesses zuweisen:

```
hugo:/usr/lib/cgi-bin# chmod go-r .htpasswd
hugo:/usr/lib/cgi-bin# chown www-data:www-data .htpasswd
hugo:/usr/lib/cgi-bin#
```

2. Dann liegt die Datei in einem Bereich, der mit Hilfe des Apache-Servers ausgelesen werden kann. Die Datei ließe sich z. B. über die Adresse `http://localhost/awstats/.htpasswd` herunterladen. Das müssen wir ebenfalls verhindern. Wir werden das bei der Anpassung der Apache-Konfiguration im Gedächtnis behalten.

Jetzt müssen wir den Bereich, in dem das CGI-Script `awstats.pl` liegt, so schützen, dass nur über die in der Datei `.htpasswd` definierten Kennungen und Passwörter darauf zugegriffen werden kann. Dazu werden wir die Datei `/etc/apache2/conf.d/awstats.conf` ein wenig erweitern:

Listing 9.9: /etc/apache2/conf.d/awstats.conf

```
                                        ...
ScriptAlias /awstats/ /usr/lib/cgi-bin/

<Directory /usr/lib/cgi-bin>
AuthType Basic
AuthName "Zugriffsstatistik"
AuthUserFile "/usr/lib/cgi-bin/.htpasswd"
Require valid-user
</Directory>
                                        ...
```

Das Beispiel hat einige Eigenheiten. Beginnen wir mit den Konfigurationsoptionen:

- `AuthType` definiert eine spezielle Art der Autorisierung. In diesem Fall wählen wir mit `Basic` das einfachste Modul, das auch i. d. R. in den Apache-Server einkompiliert ist und nicht vorher geladen werden muss.

- Mit `AuthName` legen Sie fest, was in der Dialogbox, die Kennung und Passwort abfragt, als Titel für den Bereich ausgegeben wird, für den man sich autorisieren muss.
- `AuthUserFile` gibt die Datei an, in der die Kennungen und Passwörter gespeichert werden. Hier sollten Sie immer einen vollständigen Pfad angeben. Fehlt der »/« am Anfang des Pfads, wird die Datei relativ zum `ServerRoot` gesucht, was mitunter ziemlich woanders sein kann.
- `Require` legt schließlich fest, was notwendig ist, um Seiten aus diesem Verzeichnis abzurufen. Hier bedeutet `valid-user`, dass alle in der Datei angegebenen Kennungen nach Eingabe des (richtigen) Passworts Zugriff auf die Seite erhalten. Sie können den Zugriff auch nur für bestimmte Kennungen erlauben, wenn Sie statt `valid-user` das Schlüsselwort `user` und dahinter einzelne Kennungen angeben.
- Zuletzt sei darauf hingewiesen, dass die Autorisierung entweder in einer `Directory`-Angabe oder in einer `htaccess`-Datei stehen muss. An anderer Stelle ist sie nicht erlaubt. Daher brauchen wir auch ein Verzeichnis auf der Dateisystemebene als Basis für unseren geschützten Bereich.

Ein Problem haben wir jetzt noch übrig: die Einschränkung des Downloads unserer Passwortdatei. Dazu existiert die Direktive `Files`, mit der Zugriffsbeschränkungen für Dateien definiert werden können. Diese Direktive kann auch innerhalb einer `Directory`-Direktive angegeben werden, um Zugriffe auf Dateien in diesem Verzeichnisbereich zu spezifizieren:

Listing 9.10: /etc/apache2/conf.d/awstats.conf

```
...

<Directory /usr/lib/cgi-bin>
AuthType Basic
AuthName "Zugriffsstatistik"
AuthUserFile "/usr/lib/cgi-bin/.htpasswd"
Require valid-user

<Files ".htpasswd">
Order Deny,Allow
Deny from all
</Files>

</Directory>

...
```

Mit dieser Konfiguration werden alle Versuche abgelehnt, die Datei `.htpasswd` herunterzuladen.

Konfigurationen für Verzeichnisse in den Verzeichnissen selbst speichern: Wenn wir es uns genau überlegen, haben wir hier einen kleinen Design-Fehler gemacht. Wir haben eine Konfiguration, die auf Dat(ei)en aufbaut, die sich im Verzeichnis

selbst befinden, in einer »weit entfernten« Konfigurationsdatei abgelegt. Wenn wir die dazugehörige Konfiguration in einer Datei dort abspeichern würden, wo wir sie brauchen, hätten wir alle wichtigen Daten an einer Stelle versammelt.

Auch dafür gibt es eine spezielle Funktion: Mit der Direktive `AccessFileName` wird in der `apache2.conf` festgelegt, welche Dateien zusätzlich ausgewertet werden sollen. Wenn dann für diesen Verzeichnisbereich noch mit der Direktive `AllowOverride` diese Funktion freigegeben ist, können in Dateien, die i. d. R. den Namen `.htaccess` tragen, zusätzliche Konfigurationen untergebracht werden.

In unserem Fall muss zuerst für das Verzeichnis `/usr/lib/cgi-bin` die Möglichkeit freigegeben werden, die zusätzlichen Access-Dateien nutzen zu können. Dazu werfen wir noch einmal einen Blick in unsere Site-Konfiguration `default`. Dort haben wir die Möglichkeiten im Verzeichnis rigoros eingeschränkt, was wir jetzt etwas aufweichen wollen:

Listing 9.11: /etc/apache2/sites-available/default

```
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
<Directory "/usr/lib/cgi-bin">
AllowOverride AuthConfig Limit
Options +ExecCGI -MultiViews
Order allow,deny
Allow from all
</Directory>
```

Wir ersetzen `None` durch `AuthConfig` und `Limit`, um zum einen die Autorisierung konfigurieren zu können und um unsere `Files`-Direktive unterbringen zu können. Jetzt können wir im Verzeichnis `/usr/lib/cgi-bin` eine Datei mit Namen `.htaccess` anlegen und die Konfigurationen dort hineinkopieren, die bisher in `/etc/apache2/conf.d/awstats.conf` untergebracht waren.

Listing 9.12: /usr/lib/cgi-bin/.htaccess

```
AuthType Basic
AuthName "Zugriffsstatistik"
AuthUserFile "/usr/lib/cgi-bin/.htpasswd"
Require valid-user
<Files ".ht*">
Order Deny,Allow
Deny from all
</Files>
```

Ist Ihnen die kleine Änderung zur ursprünglichen Konfiguration aufgefallen? Wir haben in der `Files`-Direktive den Platzhalter »*« eingesetzt, um unsere beiden `.ht`-Dateien vom Herunterladen auszuschließen. Haben Sie die Konfigurationsoptionen aus der Datei `awstats.conf` entfernt und dem Apache-Server mit `apache2ctl graceful` mitgeteilt, dass wir an der Konfiguration etwas geändert haben? Dann sollte sich

am Verhalten beim Aufruf der Adresse `http://docserver/awstats/awstats.pl` nichts geändert haben. ;-)

Systempasswörter mit der `htpasswd`-Datei synchronisieren:
Sowohl die Datei `/etc/passwd` als auch die heutzutage für das Speichern des Passworts genutzte Datei `/etc/shadow` kann (und sollte) vom Apache-Server nicht direkt gelesen und interpretiert werden. Dies liegt neben Sicherheitsaspekten an den in diesen Dateien enthaltenen Zusatzfeldern; Sie können aber Kennung und Passwort aus dieser Datei extrahieren und in einer separaten Datei speichern, die dann für die Authentisierung durch den Apache-Server genutzt werden kann.

Bei der »Konvertierung« hilft ein einfaches awk-Script:

```
hugo:~# gawk 'BEGIN { FS = ":"; OFS = ":"; } { if (length($s) > 1) } print $1,$2; } }'
< /etc/shadow > htpasswd.test
hugo:~#
```

Dieses sortiert aus der Datei `/etc/shadow` (oder auch aus der Datei `/etc/passwd`) die Kennungen heraus, zu denen auch ein Passwort gespeichert ist, und schreibt sie mit den dazugehörigen Passwörtern in die Datei `htpasswd.test`. Um diese aktuell zu halten, können Sie den gawk-Aufruf auch in einem Cron-Job regelmäßig ausführen. So werden die Passwortänderungen, die die Systemnutzer vornehmen, automatisch dem Apache-Server zugänglich gemacht.

Der bessere Weg: *eine* Quelle für Kennungen und Passwörter
Dies lässt sich z. B. relativ einfach erreichen, wenn auf dem System (oder im lokalen Netzwerk) die Zugangsdaten bereits durch einen zentralen LDAP-Server verwaltet werden.

Nehmen wir einmal an, wir hätten einen lokalen LDAP-Server, der als Basis-Domain `dc=Beispiel,dc=de` hat. Dann können wir mit der Direktive `AuthLDAPURL` die Autorisierung über den LDAP-Server laufen lassen:

```
AuthType Basic
AuthName "Zugriffsstatistik"
AuthLDAPURL ldap://localhost:389/dc=Beispiel,dc=de?uid
Require valid-user
```

Die Angabe `?uid` gibt an, dass nach der `UserID` gefragt werden soll. Sie können aber auch mit `?cn` den im Feld `Canonical Name` in der LDAP-DAtenbank gespeicherten Wert abfragen. Nehmen wir an, unser Nutzer hat die UserID `victor` und trägt den schönen Namen `Victor Y. Secosbosque`, der auch im `cn`-Feld eingetragen ist. Mit der `AuthLDAPURL ldap://...?uid` müsste er sich mit `victor` und seinem Passwort anmelden, bei `ldap://...?cn` müsste er `Victor Y. Secosbosque` und sein Passwort eingeben.

Die `Require`-Direktive kennt in diesem Zusammenhang zusätzliche Funktionen: So lassen sich die Kennungen, die auf die Ressource Zugriff haben sollen, mit Hilfe anderer LDAP-Attribute filtern. Es wäre z. B. möglich, mit `Require ldap-attribute`

`employeeType=Manager` nur denjenigen Kennungen den Zugriff zu erlauben, die im LDAP-Directory als »Manager« geführt sind.

Neben Autorisierung über einen LDAP-Server lassen sich aber auch andere Quellen wie z. B. Datenbanken für die Autorisierung nutzen. Wer die doch etwas komplizierte Einrichtung eines LDAP-Servers scheut, kann auch mit den Apache-Modulen `mod_auth_mysql` oder `mod_auth_pgsql` arbeiten, die von Drittanbietern zur Verfügung gestellt werden. Für die Systemebene existieren äquivalente PAM-Module, die dann die gleiche Datenbasis nutzen können.

9.6 Sicherer Zugriff mit SSL

Mittlerweile sind wir soweit, dass sensible Daten zwischen Client und Server ausgetauscht werden; sobald Passwörter durch das Netzwerk wandern, sollte man dafür sorgen, dass die Übertragung gegen das Mithören gesichert wird. Für das `http`-Protokoll hat sich zu diesem Zweck die SSL-Verschlüsselung etabliert. Die dafür notwendigen Zertifikate sind jedoch nicht auf die Nutzung durch den Apache-Server beschränkt. Ein Zertifikat kann durchaus von mehreren Anwendungen genutzt werden. Trotzdem haben wir uns entschlossen, die Nutzung von SSL-Zertifikaten am Beispiel des Apache-Servers zu erklären, da sie in diesem Umfeld am häufigsten eingesetzt werden.

Die verschlüsselte Übertragung zwischen Client und Server basiert auf einem mehrstufigen Verfahren: Der Datenaustausch wird am Ende mit einem symmetrischen Schlüssel kodiert, mit dem sowohl ver- als auch entschlüsselt werden kann. Dieser Schlüssel muss jedoch vor dem Übermitteln der eigentlichen Daten zwischen Client und Server ausgetauscht werden. Dies geschieht mit Hilfe eines asymmetrischen Verfahrens: Der Client verschlüsselt den symmetrischen Schlüssel mit dem öffentlichen Schlüssel des Servers, den dieser dann mit seinem privaten Schlüssel

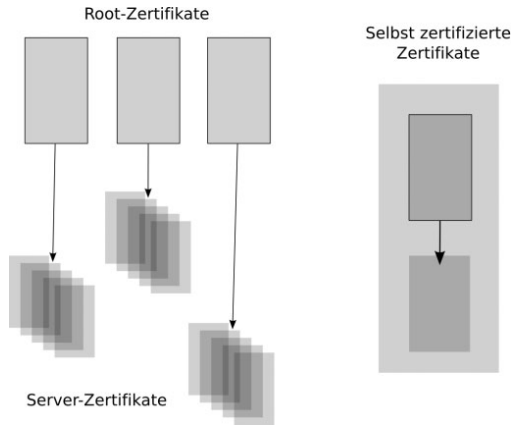


Abbildung 9.5: Zertifikate

entschlüsselt und für die weitere Verschlüsselung nutzt. Wie gelangt der Client nun an den öffentlichen Schlüssel des Servers? Der organisatorische Aufwand wäre zu groß, auf allen Clients die öffentlichen Schlüssel aller im Internet zur Verfügung stehenden Server vorzuhalten.

Der Server übermittelt also als Erstes den öffentlichen Teil seines Schlüssels dem Client. Wir haben also zu Beginn des Datenaustauschs eine unsichere Phase, in der der Client den Schlüssel des Servers herunterlädt. Der Client benötigt eine Möglichkeit zu kontrollieren, ob Server und Schlüssel zusammengehören. Danach erst folgt die sichere Phase mit verschlüsseltem Datenverkehr.

Damit der Client kontrollieren kann, ob der übermittelte Schlüssel sicher ist und ihm nicht ein falscher Schlüssel untergejubelt wird, wird der Server-Schlüssel von einem sogen. root-Zertifikat signiert. Dieses gehört einer **C**ertificate **A**uthority (CA) und wird i. d. R. bei der Auslieferung des Clients mitgeliefert. Der Client besitzt also *vor* dem Empfang des Schlüssels vom Server ein Werkzeug, um die Echtheit des übermittelten Schlüssels zu kontrollieren.

Bei der Beantragung eines Zertifikats bei einer CA müssen Sie neben Informationen zu Ihnen und Ihrer Organisation auch den Namen des Servers angeben, für den das Zertifikat bestimmt ist. Stimmt der Name des Servers, von dem das Zertifikat heruntergeladen worden ist, nicht mit dem im Zertifikat gespeicherten Namen überein, meldet der Client dies dem Nutzer und/oder bricht die Verbindung ab.

Achtung

Merke: Zertifikate sind an eine IP-Nummer gebunden. Wenn Sie auf verschiedenen Servern SSL-Verbindungen nutzen wollen, müssen Sie für jeden Server ein eigenes Zertifikat benutzen!

9.6.1 Externe Zertifikate einbinden

Was ist notwendig, um bei einem Apache-Server verschlüsselte Verbindungen zu aktivieren?

- Wir benötigen grundsätzlich eine Serverkonfiguration für den HTTPS-Port 443. Diese kann sich durchaus von den anderen Konfigurationen unterscheiden und parallel zu den normalen Konfigurationen betrieben werden. Sie können aber in einer Konfiguration nicht verschlüsselten und unverschlüsselten Verkehr zusammen betreiben. Daher ist ein separater virtueller Host für den SSL-Verkehr eine sinnvolle Lösung.
- Dann brauchen wir ein Zertifikat
- und müssen es in der Apache-Konfiguration einbinden.

Serverkonfiguration für den SSL-Server: Nehmen wir einmal an, wir hätten für unseren Webserver `www.beispiel.org` ein Zertifikat beantragt und wollen diesen Service auf unserem Server `hugo` anbieten. Wir brauchen also auf jeden Fall einen namensbasierten, virtuellen Host auf einem bisher noch nicht genutzten Port. Daher sollten wir die Datei `ports.conf` um Einträge für den SSL-Port 443 ergänzen:

Listing 9.13: /etc/apache2/ports.conf

```
Listen 192.168.1.10:80
NameVirtualHost 192.168.1.10:80

Listen 192.168.1.10:443
NameVirtualHost 192.168.1.10:443
```

Im nächsten Schritt legen wir eine Konfiguration an, die wir für den verschlüsselten Datenverkehr nutzen wollen. Sie haben wahrscheinlich schon eine Reihe virtueller Hosts konfiguriert? Suchen Sie sich diejenige Konfiguration, die der neuen Variante am nächsten kommt und kopieren Sie die Datei. Wir nehmen dafür unsere `docserver`-Konfiguration und kopieren sie mit dem Namen `443default`.

Listing 9.14: /etc/apache2/sites-available/443default

```
<VirtualHost 192.168.1.10:443>
        ServerName www.beispiel.org
        ServerAdmin victor@www.beispiel.org
                                            ...
</VirtualHost>
```

Die wichtigen Änderungen sind die Angabe des SSL-Ports `443` und der Name des Servers `www.beispiel.org`, für den das Zertifikat beantragt worden ist.

Wohin mit dem Zertifikat? Sie erhalten i. d. R. eine Datei mit der Endung `.crt`, die einen privaten Schlüssel und das Zertifikat enthält. Diese sind in kodierter Form gespeichert und nicht direkt lesbar. Sie finden in der Datei aber Kopf- und Fußzeilen, die den jeweiligen Bereich einklammern:

Listing 9.15: /etc/apache2/ssl/www.crt

```
-----BEGIN RSA PRIVATE KEY-----
AoGAHbkQyxh7rfmBJnXq17DAFtnGCExhI4NAQ7+7mz5O5WoAeOr5u7V1LfRW4yYG
                                            ...
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
oaEoNZRqiO5MmIaFOiQVoZWZvNW4iHXq7SzuNocmyYO9gS/B5o6FWwcM/YC+OcRy
                                            ...
-----END CERTIFICATE-----
```

Wir haben für unseren Schlüssel ein Verzeichnis `ssl` im Konfigurationsbereich unseres Webservers angelegt und die Datei dort gespeichert. Bevor wir dieses Zertifikat

dem Webserver zugänglich machen, ist noch etwas Voodoo notwendig. Die Prozesse, die auf einen Schlüssel zugreifen, hätten gerne einen Dateinamen, der dem sogenannten Hashwert der Datei entspricht. Diesen werden wir jetzt noch als symbolischen Link auf unsere Schlüssel-Datei erzeugen:

```
hugo:/etc/apache2/ssl# ll
-rw------- 1 root root 2,0K nnnn-nn-nn nn:nn www.crt
hugo:/etc/apache2/ssl# ln -s www.crt $(openssl x509 -hash -noout -in www.crt)
hugo:/etc/apache2/ssl# ll
lrwxrwxrwx 1 root root    8 nnnn-nn-nn nn:nn 765g3480 -> lena.crt
-rw------- 1 root root 2,0K nnnn-nn-nn nn:nn www.crt
hugo:/etc/apache2/ssl#
```

Damit sind alle Vorbereitungen getroffen, um unseren Apache-Server mit Super-Kuh-Kräften auszustatten.

SSL-Verschlüsselung aktivieren: Zuerst kontrollieren wir, ob das SSL-Modul bereits geladen ist: Dazu genügt ein Blick in das Verzeichnis `mods-enabled`. Sie sollten dort die symbolische Links `ssl.load` und `ssl.conf` finden.

```
hugo:/etc/apache2# ll mods-enabled
[...]
lrwxrwxrwx [...] ssl.conf -> /etc/apache2/mods-available/ssl.conf
lrwxrwxrwx [...] ssl.load -> /etc/apache2/mods-available/ssl.load
[...]
hugo:/etc/apache2#
```

Diese enthalten einige Konfigurationsoptionen, die für die »Selbstorganisation« des Apache-Servers notwendig sind, aber nicht spezifisch für ein Zertifikat sind. Wir wollen an dieser Stelle nicht näher auf diese Optionen eingehen, weil die Voreinstellungen für diese Module i. d. R. sinnvoll gewählt sind und nicht unbedingt angepasst werden müssen.

Fehlen die symbolischen Links im Verzeichnis `mods-enabled`, können Sie die Module mit `a2enmod ssl` aktivieren. In der Regel ist SSL nach der Installation aber aktiviert – insbesondere wenn Sie bei Ihrer Distribution dafür die Module als separates Paket nachinstallieren mussten. Jetzt muss SSL nur noch in der Konfigurationsdatei angeschaltet werden. Dies wird durch die Direktiven `SSLEngineOn`erreicht; außerdem muss dem Server mit der Direktive `SSLCertificateFile` mitgeteilt werden, in welcher Datei sein Schlüssel gespeichert ist:

Listing 9.16: /etc/apache2/sites-available/443default

```
<VirtualHost 192.168.1.10:443>
        ServerName www.beispiel.org
        ServerAdmin victor@www.beispiel.org
```

```
        SSLEngine On
        SSLCertificateFile /etc/apache2/ssl/www.crt
                                                ...
</VirtualHost>
```

Jetzt können Sie mit `/etc/init.d/apache2 reload` den Apache-Server neu starten und dann einen Versuch unternehmen, über eine Adresse der Form `https://www.beispiel.org` eine Seite abzurufen.

9.6.2 Selbst zertifizierte Zertifikate generieren

Zertifikate sind eine teure Angelegenheit, weil die Firmen, die diese anbieten, sich den organisatorischen und sicherheitstechnischen Aufwand mit regelmäßigen Obuli bezahlen lassen. Für den internen Gebrauch kann eine verschlüsselte SSL-Verbindung aber auch ohne die Sicherheit einer großen Certificate Authority nützlich sein. Sie können daher auch selbst Zertifikate generieren, die für eine Verschlüsselung genutzt werden können. Wir zeigen Ihnen gleich den einfachen Weg mit Hilfe selbst zertifizierter Zertifikate.

Sie können auch einen sichereren Weg wählen, indem Sie ein eigenes CA-Zertifikat erzeugen, mit dem Sie dann die Server-Zertifikate signieren. Den öffentlichen Teil Ihres CA-Zertifikats können Sie dann den Clients mitgeben oder zum Download anbieten. Ist das CA-Zertifikat einmal im Client »installiert«, werden alle damit signierten Server-Zertifikate ohne Nachfrage akzeptiert. Genauere Informationen dazu finden Sie im »SSL-Certificates-HOWTO«.

Selbst zertifizierte Zertifikate generieren: Zertifikate lassen sich mit einem Aufruf des Programms openssl generieren. Zertifikate gelten i. d. R. nur für einen festgelegten Zeitraum. Sie können sich die Arbeit erleichtern, wenn Sie die Angaben zum Zertifikat in einer Konfigurationsdatei speichern und diese beim Generieren eines Nachfolgezertifikats wiederverwenden.

Wenn Sie die Software openssl installieren, wird i. d. R. ein Verzeichnis `/etc/ssl` angelegt, in dem Sie eine Datei `openssl.cnf` finden. Darin sind Einträge der Form `countryName` oder `emailAddress` enthalten. Wenn Sie ein neues Zertifikat generieren, fragt openssl einige Parameter ab und gibt die hinter diesen Schlüsselwörtern gespeicherten Texte als Eingabeaufforderung aus. Sie können jetzt Ihre Antwort vorbelegen, indem Sie in einer neuen Zeile eine Konfigurationsoption ergänzen, die den jeweiligen Namen, ergänzt um den Text `_default`, enthält:

Listing 9.17: /etc/ssl/openssl.cnf

```
                                        ...
commonName                      = Common Name (eg, YOUR name)
commonName_max                  = 64
commonName_default = Victor Y. Secosbosque
```

```
emailAddress                    = Email Address
emailAddress_max                = 64
emailAddress_default            = victor@hugo.beispiel.org
                                        ...
```

Alle Einträge, die Sie hier vorgeben, müssen Sie später bei der Erzeugung des Zertifikats nur noch mit der Eingabetaste bestätigen.

Kommen wir nun zu der Erzeugung des Zertifikats. Auch hier haben wir einen Voodoo-ähnlichen Aufruf des openssl-Programms auszuführen:

```
hugo:/etc/apache2/ssl# openssl req -config /etc/ssl/openssl.cnf -new -x509 -nodes -days 3000
-out lokal.crt -keyout lokal.crt
Generating a 1024 bit RSA private key
...++++++
.....................................................++++++
writing new private key to 'lokal.crt'
-----
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.
-----
Country Name (2 letter code) [DE]:
                                        ...
hugo:/etc/apache2/ssl# openssl x509 -text -in lokal.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            8b:a5:30:1b:3d:5c:4e:14
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: [...]
        Validity
            Not Before: Mar 20 20:41:49 2006 GMT
            Not After : Jun  6 20:41:49 2014 GMT
                                        ...
hugo:/etc/apache2/ssl# ln -s lokal.crt $(openssl x509 -hash -noout -in lokal.crt)
hugo:/etc/apache2/ssl#
```

Nach dem Aufruf generiert openssl zuerst einen Schlüssel. Danach fragt das Programm für die Erstellung des Zertifikats einige Daten von Ihnen ab. Die von Ihnen voreingestellten Parameter finden Sie in den eckigen Klammern. Mit Return können Sie die Vorgabe übernehmen. Am Ende hat openssl die unter `-out` und `-keyout` an-

gegebene Datei erzeugt. Achten Sie darauf, dass Sie zweimal den gleichen Namen angeben!

Auf eine Option wollen wir Sie noch hinweisen: Mit `-days xxx` legen Sie fest, wie lange dieses Zertifikat gültig ist. Nach Ablauf dieses Zeitraums bekommen die Nutzer beim Aufruf von Webseiten dieses Servers eine Fehlermeldung, dass das Zertifikat nicht mehr gültig ist. Legen Sie hier also einen Rhythmus fest, an den Sie sich gut erinnern können.

Mit dem Kommando `openssl x509 -text -in <Zertifikat>` können Sie sich den Inhalt einer Zertifikat-Datei ausgeben lassen. Dort finden Sie neben den Daten zur Person und Organisation auch den Zeitraum, den das Zertifikat gültig ist. Und vergessen Sie zum Schluss nicht das Hash-Voodoo, damit der Apache-Server nicht unnötig im Verzeichnis herumtanzt und das Zertifikat sucht!

Welche Sicherheit haben wir mit einem selbst zertifizierten Zertifikat erreicht?

Das Zertifikat erlaubt Ihnen die Nutzung einer verschlüsselten Übertragung, nicht mehr und nicht weniger. Was Ihnen fehlt, ist die Autorisierung des Zertifikats durch eine übergeordnete Stelle. Damit kann der Client nicht kontrollieren, ob das Zertifikat wirklich von Ihrem Server kommt oder jemand eine »Man-in-the-middle«-Attacke durchführen will. Davor können Sie sich nur schützen, wenn Sie den Zertifikatsteil vorher den Clients zur Verfügung stellen.

- **Der einfache Weg:** Sie rufen die Webseite auf und bestätigen das Zertifikat. Dies können Sie temporär oder dauerhaft tun, wobei Sie aber die Gefahr einer »Man-in-the-middle«-Attacke in Kauf nehmen. Wie realistisch das ist, müssen wir Ihrer Einschätzung überlassen.
- **Austausch des Zertifikats:** Bei wenigen Clients und einem Server können Sie den Zertifikatsteil der `crt`-Datei in eine eigene Datei speichern. In unserem Beispiel würden wir diese Datei `lokal.pem` nennen. Diese können Sie auf den Clients als Zertifikat importieren. Die Vertrauenswürdigkeit bestimmen in diesem Fall Sie durch den Weg, auf dem das Zertifikat zu den Clients gelangt.
- **Eine eigene CA:** Sobald Sie mehrere Server mit einem sicheren Zugriff ausstatten wollen, lohnt sich die Einrichtung eines root-Zertifikats. Sie müssen zwar weiterhin dafür sorgen, dass dieses auf einem sicheren Weg zu Ihren Clients gelangt; es reicht aber weiterhin ein Zertifikat aus, um alle damit signierten Server zu autorisieren.

9.6.3 SSL: Ein weites Feld ...

Wir haben die Möglichkeiten, die eine SSL-Verschlüsselung im Allgemeinen und openssl im speziellen bietet, nur anreißen können. So lassen sich z. B. Zertifikate für die Verschlüsselung anderer Internet-Protokolle wie Mail oder für die Verschlüsselung von Dateien nutzen. Für eine genauere Darstellung wäre selbst ein ganzes Buch notwendig. Daher möchten wir Ihnen an dieser Stelle noch ein paar Literaturhinweise geben, die Sie bei Interesse weiterbringen:

Weitere Literatur

- »SSL Certificates HOWTO«: http://www.tldp.org/HOWTO/SSL"=Certificates"=HOWTO/
- OpenSSL-Homepage: www.openssl.org

9.7 Apache als Fileserver mit Hilfe von WebDAV

Nun haben wir uns lang und breit über das Herunterladen ausgelassen; da wird es Zeit, sich über das Hochladen Gedanken zu machen: Mit Hilfe des Protokolls WebDAV kann der Apache-Server nämlich auch mit dem Upload von Dateien umgehen. Die Grundlage dafür sind die Module `mod_dav` und `mod_dav_fs`. Ersteres stellt die grundsätzlichen WebDAV-Funktionen zur Verfügung, während letzteres sozusagen das ausführende Organ darstellt und die Funktionen im Dateisystem realisiert.

```
hugo:/etc/apache2# a2enmod dav
Module dav installed; run /etc/init.d/apache2 force-reload to enable.
hugo:/etc/apache2# a2enmod dav_fs
Module dav_fs installed; run /etc/init.d/apache2 force-reload to enable.
hugo:/etc/apache2# /etc/init.d/apache2 force-reload
Forcing reload of apache 2.0 web server....
hugo:/etc/apache2#
```

Was ist WebDAV oder kurz DAV? Die Abkürzung steht für **D**istributed **A**uthoring and **V**ersioning. DAV ist also deutlich mehr als nur eine Art FTP-Erweiterung für das HTTP-Protokoll. Unter Windows lassen sich DAV-Ordner als Web-Ordner einbinden; dies ist insbesondere deswegen nützlich, weil Tools wie ssh oder sftp unter Windows noch nicht in dem Maße unterstützt werden wie unter Linux. DAV in Verbindung mit SSL stellt hier eine sichere und einfach zu nutzende Alternative dar.

Serverseitig kann damit Software realisiert werden, die ein gemeinsames Arbeiten an Dokumenten unterstützt. Dazu gehören Features wie das Locking auf Dateibasis, sodass eine Datei, während sie von einem Nutzer bearbeitet wird, nicht von einem anderen Nutzer unbeabsichtigt überschrieben werden kann. Außerdem können zu Dateien Metadaten gespeichert werden, wie z. B. Autorenlisten u.Ä. So arbeitet das Versionsverwaltungssystem Subversion mit einem DAV-Modul für den Apache-Server, um den Up- und Download von Dateien zu ermöglichen. Das Modul `dav_fs` realisiert in diesem Zusammenhang aber nur die Basis-Funktionen des Hochladens, Kopierens und Löschens.

Nehmen wir einmal an, wir wollen ein Verzeichnis `Downloads`, das einen Download-Bereich für PDF-Dokumente enthält, auf unserem Webserver für den Upload zugänglich machen. Nachdem wir die DAV-Module geladen haben, können wir eine `Directory`-Direktive um DAV-Funktionen erweitern:

```
Alias /Downloads /srv/www/downloads
<Directory /srv/www/downloads>
```

```
Options +Indexes -ExecCGI
Order Deny,Allow
Allow from all

Dav on
<LimitExcept GET OPTIONS>
AuthType Basic
AuthName ''PDF-Uploadbereich''
AuthUserFile /etc/apache2/authfiles/pdf_upload.passwd
Require valid-user
</Limit>

</Directory>
```

Mit der Option `Dav on` wird DAV mit dem Modul `dav_fs` aktiviert. Wenn Sie serverseitig Module von Drittanbietern nutzen, müssen Sie statt `on` den Namen des Moduls angeben, da `on` nur ein Alias für `dav_fs` darstellt. Das wäre es eigentlich gewesen, wenn wir uns nicht noch Gedanken darüber machen wollen, wer denn jetzt etwas hochladen darf und wer nicht.

Da wir bisher für unser Verzeichnis keine Einschränkung definiert haben, könnte jeder, der auf die Webseite und das Verzeichnis zugreifen kann, Dateien hochladen, bewegen oder löschen. Das ist aber nicht der Sinn der Sache. Daher schränken wir den freien Zugriff mit `LimitExcept` auf die HTTP-Kommandos `GET` und `OPTIONS` ein. Diese müssen auf frei zugänglichen Webseiten allen Nutzern erlaubt sein, um Seiten herunterladen zu können. Alle anderen `HTTP`-Kommandos wie z. B. das für das Hochladen notwendige `PUT` werden nur nach Autorisierung zur Verfügung gestellt.

Da hier für den Upload Kennungen und Passwörter durch das Netz laufen, sollten Sie diese Konfiguration im SSL-Bereich Ihres Webservers unterbringen bzw. nur dort aktivieren. Wenn Sie einen Bereich komplett nur nach Anmeldung zur Verfügung stellen wollen, lassen Sie, wie oben schon beschrieben, die `LimitExcept`-Direktive einfach weg.

Achtung

Planen Sie den Einsatz von DAV sorgfältig! Es lässt sich für Unterverzeichnisse nicht mehr deaktivieren! Einmal freigegeben, alles freigegeben.

Was passiert beim Hochladen von Dateien? Damit Dateien hochgeladen, Verzeichnisse angelegt und Dateien bewegt oder gelöscht werden können, muss die

Kennung, unter der der Apache-Server läuft, Schreibrechte auf diesen Verzeichnisbereich haben. Alle hochgeladenen Dateien werden dann mit der Kennung des Apache-Servers im Dateisystem angelegt. Das garantiert zum einen, dass alle Kennungen, die auf diesen Bereich zugreifen können, auch mit allen Dateien alles machen können. Zum anderen zeigt es, dass Sie jetzt keine speziellen Kennungen mehr auf Dateisystemebene benötigen und die Zugriffsrechte über `htpasswd`-Dateien abwickeln können. Unnötig zu sagen, dass natürlich alle Autorisierungsarten auch mit DAV-Bereichen funktionieren, Sie also auch z. B. LDAP oder eine datenbankgestützte Autorisierung nutzen können.

Obiges Szenario können Sie z. B. auch auf Bereiche anwenden, in die Nutzer Webseiten mit Programmen wie DreamWeaver oder FrontPage hochladen. Den Download, also das Ansehen der Seiten, schalten Sie frei; für das Hochladen (über eine SSL-Verbindung ;-)) muss ein Passwort eingegeben werden. Dies funktioniert mit statischen HTML-Seiten problemlos, wie sie von Webauthoring-Software erstellt werden. Ein Problem ergibt sich jedoch bei dynamischen Webseiten, die vor dem Herunterladen vom Server geparst, durch eine Engine wie z. B. PHP gejagt oder wie bei CGI-Scripts ausgeführt werden.

In einem solchen Fall sollten Sie DAV über eine Location-Direktive aktivieren und einen zweiten Weg zu den gleichen Dateien ermöglichen:

```
Alias /phpbereich /srv/www/phpbereich
<Directory /srv/www/phpbereich>
Order Deny,Allow
Allow from all
</Directory>
Alias /phpsource /srv/www/phpbereich
<Location /phpsource>
   Dav on
   ForceType text/plain
</Location>
```

Hier bedienen wir uns zweier Tricks: Zum einen kann ein `Alias` durchaus auf ein bereits mit einem anderen Alias zur Verfügung gestelltes Verzeichnis verweisen. Ohne die `Location`-Direktive bekäme man unter `/phpsource` das gleiche zu sehen wie unter `/phpbereich`. Da der Apache-Server aber anhand des Dateityps entscheidet, was er mit den Dateien anfängt, werden außerhalb der `Location`-Direktive die PHP-Dateien an die PHP-Engine weitergegeben und damit ausgeführt. Für `phpsource` haben wir jedoch mit `ForceType` die Behandlung aller Dateien als Textdateien erzwungen, wodurch alle Dateien ohne Spezialbehandlung ausgeliefert werden. Über die Adresse `/phpsource` können die PHP-Seiten jetzt heruntergeladen, bearbeitet und wieder hochgeladen werden.

Weitere Literatur

- Dokumentation zum Modul `mod_dav`
- Webseiten des DAV-Projekts: `webdav.org`

9.8 Dynamische Webseiten

Bisher haben wir hauptsächlich über Webseiten oder Dateien gesprochen, die so ausgeliefert werden, wie sie auf dem Server liegen. Heutzutage werden jedoch viel Websites dynamisch auf dem Server aus Einzelteilen zusammengesetzt oder entstehen durch Zugriffe auf Datenbanken. Das bedeutet i. d. R., dass beim Aufruf einer Seite ein Script oder ein Programm ausgeführt wird, das eine HTML-Seite generiert und zurückliefert.

Programme ausführen über das Common Gateway Interface: Eine Schnittstelle zu Programmen haben wir weiter oben schon gezeigt: die sogenannten CGI-Scripts. Kurz zusammengefasst wird ein Verzeichnis als CGI-Verzeichnis freigegeben; alle darin liegenden, ausführbaren Dateien werden als Programme aufgefasst und beim Abruf ausgeführt. Je nach Typ des Programms wirft eine solche Verarbeitung Probleme auf: Handelt es sich bei den Dateien z. B. um Perl-, PHP- oder Python-Scripts, muss für jeden Seitenabruf die jeweilige Engine gestartet und dann das Script abgearbeitet werden. Dadurch verlangsamt sich zum einen die Auslieferung der Seiten und andererseits wird der Server bei vielen Seitenabrufen durch die zahlreichen Programmstarts stark belastet.

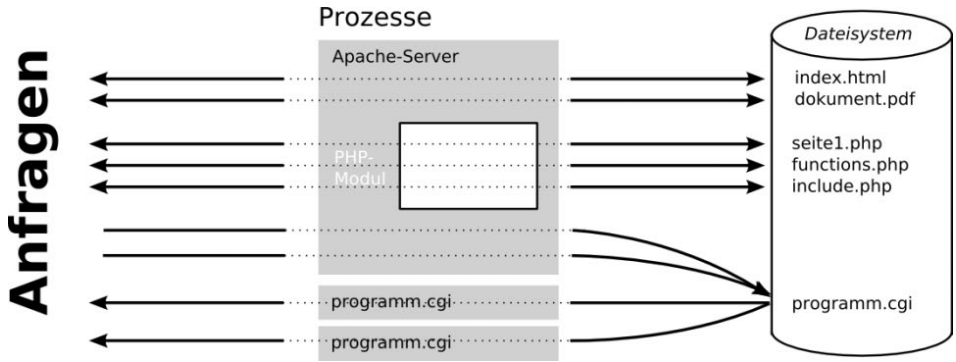


Abbildung 9.6: CGI-Zugriffe

Ein Lösungsweg für dieses Problem ist die Integration der Scriptumgebungen als Modul in den Apache-Server. In diesem Fall ist die notwendige Engine bereits geladen, und bei jedem Seitenabruf muss nur das jeweilige Script abgearbeitet werden. Daß heutzutage viele Administratoren aber wieder auf die langsamere CGI-Variante umschwenken, liegt an Sicherheitsproblemen: Hat eines der Module bzw. eines der Scripts eine Sicherheitslücke, trifft dies je nach Konfiguration alle vom Server im Dateisystem erreichbaren Seiten und Verzeichnisse. Sind Teile davon unter der Kennung schreibbar, unter der der Server läuft, entsteht eine ernste Sicherheitslücke. Mit Hilfe des Moduls `suEXEC` können Sie CGI-Scripts unter der Kennung des Nutzers laufen lassen, der die Datei erstellt hat bzw. dem diese »gehört«.

Die Wiederauferstehung von FastCGI: Es gibt mit FastCGI schon länger eine Variante, die das Geschwindigkeitsproblem von CGI lösen kann. Jedoch hat das dafür

zuständige Apache-Modul nie ganz die Qualität und Verlässlichkeit, die für vielbesuchte Websites notwendig ist. Außerdem spielt es neben dem zum Standard gewordenen `mod_php` nur eine untergeordnete Rolle und wurde in den letzten fünf Jahren nur spärlich weiterentwickelt. FastCGI-Implementation funktionieren in anderen Webserver-Lösungen wie lighthttpd und Zeus aber durchaus gut. Was tun in einem solchen Fall? Sind andere Götter neben dem Apache zulässig?

Der Apache-Server als vorgeschalteter Proxy: In den letzten Jahren verbreitete sich ein weiteres Konzept, über das dynamisch generierte Webseiten erzeugt werden können. Webapplikationsserver wie Zope enthalten eigene HTTP-Server, die für die Applikationsumgebung optimiert sind, in der sie laufen. Hier teilen sich der Apache-Server und der Applikationsserver die Aufgaben; der Apache-Server sorgt für den »Kundenkontakt«, während der Applikationsserver sich um die Produktion der Seiten kümmert.

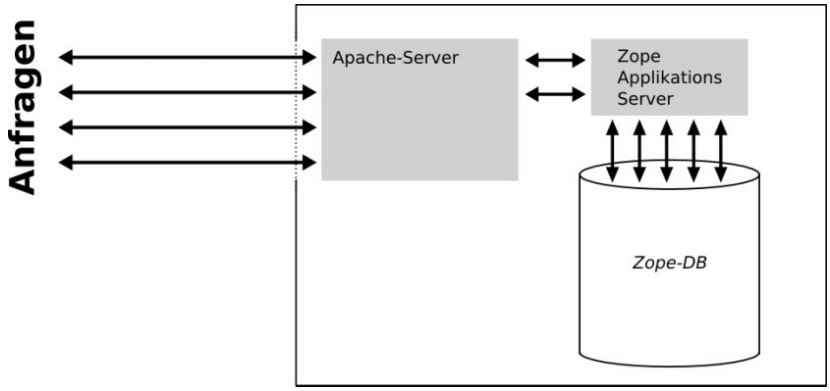


Abbildung 9.7: Webserver mit Zope

Dafür muss der Apache-Server im Proxy-Modus laufen und über das Modul `mod_rewrite` die abgerufenen Adressen passend zum Applikationsserver umschreiben. Der Zope-Server übernimmt neben der Generierung der Seiten auch die Authentisierung der Nutzer und alle anderen Aufgaben, die mit der Verwaltung der Webseiten zu tun haben. Da dies bei Zope i. d. R. in einer eigenen Datenbank abläuft, bringt eine solche Lösung mehrere Sicherheits- und Geschwindigkeitsvorteile:

- Es sind keine Systemnutzer mehr notwendig, um Schreibzugriffe auf den Webbereich zu realisieren.
- Die Auslieferung ist von der Erzeugung der Seiten getrennt: Beide Prozesse laufen mit unterschiedlichen Systemkennungen – oder sollten es zumindest. Ein Einbruch in eine der beiden Applikationen sollte keine Kompromittierung der jeweils anderen zur Folge haben.
- Beide Applikationen müssen nicht auf einem physikalischen (oder virtuellen) Server laufen. Zum Austausch der Daten zwischen Apache-Server und Zope ist nur das HTTP-Protokoll notwendig.

- Für die Generierung der Webseiten sollten mehrere Schichten zur Verfügung stehen: Je übersichtlicher der Programm-Code, desto einfacher sind die Wartung und die Fehlersuche. Zope besitzt in diesem Zusammenhang drei Ebenen: Eine einfache Scriptsprache DTML, mit der bereits komplexe Seiten und Datenbankzugriffe realisiert werden können, darunter liegt zuschaltbar der komplette Funktionsumfang der Scriptsprache Python und für bestmögliche Ausführungsgeschwindigkeit sorgt die zu Python gehörende Möglichkeit, in maschinenennahen Programmiersprachen geschriebene Teile einzubinden.

Der Apache-Server ist darauf spezialisiert, Seiten auszuliefern, der Applikationsserver darauf, Seiten zu generieren. Für Websites mit hoher Last kann jetzt an zwei Stellen ein »Fine-Tuning« vorgenommen werden: Zum einen kann man den Applikationsserver auf seine spezielle Aufgabe dressieren. Steigen die Zugriffszahlen bzw. die Last auf dem Applikationsserver, kann der Apache-Server so konfiguriert werden, dass er z. B. eine abgerufene Seite eine Minute zwischenspeichert und erst danach eine aktuelle Version vom Applikationsserver generieren lässt. Bei einer Quote von tausend abgerufenen Seiten pro Minute müsste der Applikationsserver dann nur noch jeden tausendsten Abruf neu erzeugen.

Programme haben Fehler, dynamische Webseiten auch! Wenn Sie den Pfad der statischen HTML-Seiten verlassen und sich in die Fährnisse dynamischer Webseitengenerierung begeben, sollten Sie die Komplexität dieses Themas nicht unterschätzen. Wenn Linux in der letzten Zeit aufgrund von Sicherheitslücken ins Gerede gekommen ist, waren mit überdurchschnittlicher Häufigkeit Fehler in PHP-basierten Content-Management-Systemen der Grund dafür. Bei der Entscheidung für eine Applikationsumgebung sollte man daher nicht nur die Einfachheit der Nutzung, sondern auch Sicherheitsaspekte einbeziehen. Für die Autoren sind daher weder PHP noch die Java/XML-basierten Systeme wie z. B. die Apache-eigenen Projekte Tomcat und Cocoon eine glückliche Lösung. Wenn Sie es sich leisten können, »etablierte« Standards ein Stück weit ignorieren zu können, suchen Sie sich eine Applikationsumgebung, die den oben genannten Punkte genügt und sich hinter einen Apache-Server hängen lässt. Oder Sie nutzen die Möglichkeiten von FastCGI mit einem der HTTP-Server, die dafür eine gute Unterstützung bieten. Den Apache-Server können Sie ja trotzdem davorschalten.

Weitere Literatur

- Zu suEXEC existiert ein ausführlicher Beitrag in der Apache-Dokumentation
- Zu den Stichwörtern »FastCGI«, »SCGI« und »Apache« finden Sie z. B. im Blog von Mark Majo (`vmunix.com`) einen Eintrag unter dem Titel »FastCGI, SCGI, and Apache: Background and Future«
- Zope-Homepage auf `zope.org`
- Die deutsche Usergroup zu Zope auf `dzug.org`

10 Mit SQL-Datenbanken im Netzwerk

An dieser Stelle Datenbanken für Linux ausführlich zu behandeln würde mehr als ein Buch erfordern. Daher beschränken wir uns an dieser Stelle auf die Konzepte, in denen sich netzwerkbasierte Datenbanken von den Standalone-Systemen unterscheiden. Datenbanken sind aus dem Serverbetrieb nicht mehr wegzudenken, auch wenn sie vielleicht nicht immer offensichtlich in Erscheinung treten. Die meisten PHP-basierten Content-Management-Systeme arbeiten mit einer SQL-Datenbank zusammen, i. d. R. MySQL – wir erwähnten LAMP im vorherigen Kapitel. Wir wollen Ihnen an dieser Stelle kurz die netzspezifischen Besonderheiten zweier Datenbanksysteme, MySQL und PostgreSQL, vorstellen.

10.1 Datenbanken unter Linux

Im Gegensatz zu »kleinen« Datenbanksystemen wie z. B. BerkeleyDB oder SQLite, die auf der Basis einzelner, i. d. R. in Ihrem Home-Verzeichnis gespeicherter Dateien arbeiten, funktionieren die »großen« Datenbanken nach dem Client-Server-Prinzip: Für die Nutzung der Datenbank muss ein Serverprozess laufen. Die eigentliche Datenbank wird in einem speziellen Bereich auf der Festplatte vom Server in einer oder mehreren Dateien verwaltet.

Der Zugriff auf den Datenbankserver wird, wenn sich der Client physikalisch auf dem gleichen Rechner befindet, über einen sogenannten Socket abgewickelt. Dies ist eine Datei, die beim Start des Datenbankservers angelegt wird und von den Clients beschrieben und gelesen werden kann.

Zusätzlich kann über das Netzwerk auf den Datenbankserver zugegriffen werden. Dies muss in der Konfiguration aktiviert werden, da der Netzwerkzugriff bei den meisten Distributionen standardmäßig abgeschaltet ist. Außerdem müssen die spezifischen Ports (3306 für MySQL und 5432 für PostgreSQL) in der Firewall freigegeben werden.

Datenbank-Clients können z. B. einzelne Softwaresysteme oder Bibliotheken in Scriptsprachen wie PHP oder Python sein. Der Ablauf eines Zugriffs in einer Scriptsprache läuft i. d. R. so ab, dass zuerst eine Verbindung geöffnet wird und diese dann über ein Handle von speziellen Funktionen für Lese- und Schreibzugriffe ge-

nutzt wird. Ob der Zugriff lokal über den Socket oder über das Netzwerk erfolgt, ist nur beim Öffnen der Verbindung wichtig.

Beide Datenbanksysteme bringen eine eigene Verwaltung von Kennungen und Passwörtern mit, die sehr fein auf verschiedene Zugriffsszenarien abgestimmt werden kann. Darunter fallen z. B. Regeln, welche Kennung von welchem Rechner aus auf welche Datenbank zugreifen kann etc.

Eine »Datenbank« ist ein Container, der nahezu beliebig viele Tabellen enthalten kann. Zugriffsrechte können auf die ganze Datenbank oder einzelne Tabellen vergeben werden, sodass hier fein granulierte Sicherheitsstrukturen möglich sind.

Wenn Sie den Einsatz eines Datenbank-Systems wie MySQL oder PostgreSQL als Basis für eine webbasierte Applikation planen, können Sie bei Berücksichtigung einiger Grundregeln von der Flexibilität der Systeme bezüglich des Zugriffs profitieren.

In einer **Testphase** können Sie Web-, Applikations- und Datenbankserver auf demselben System installieren (Abb. 10.1). Wenn Sie z. B. Zope nutzen, richten Sie dort einen Datenbankadapter ein, in dem die Art des Zugriffs auf die Datenbank spezifiziert ist; alle Methoden, die die eigentlichen Zugriffe durchführen, greifen auf diesen zurück. Arbeiten Sie mit PHP, sollten Sie die Zugriffsdaten zentral in einer Variable speichern, damit Sie Änderungen an der Konfiguration nur an einer Stelle vornehmen müssen.

Gehen Sie mit der Applikation in die **Produktionsphase**, können Sie die Datenbank auf einen separaten Server verlagern, wenn Sie befürchten, dass der eine Server mit dieser Aufgabe überlastet ist (Abb. 10.2). Die einzige Änderung, die Sie an Ihrer Applikation vornehmen müssen, ist die Spezifizierung des separaten Servers als Quelle für die Datenbankzugriffe – in Zope z. B. wäre nur eine Änderung im

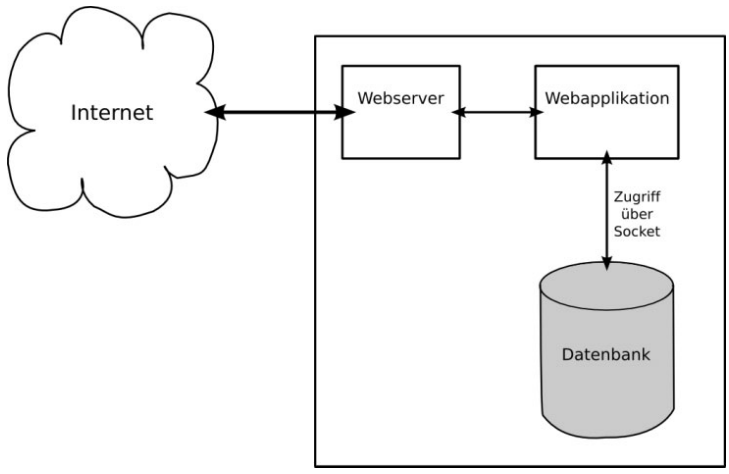


Abbildung 10.1: Datenbankanbindung an einen Webserver

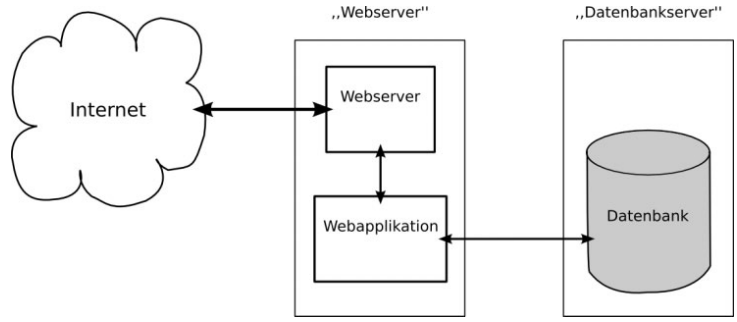


Abbildung 10.2: Separater Datenbankserver

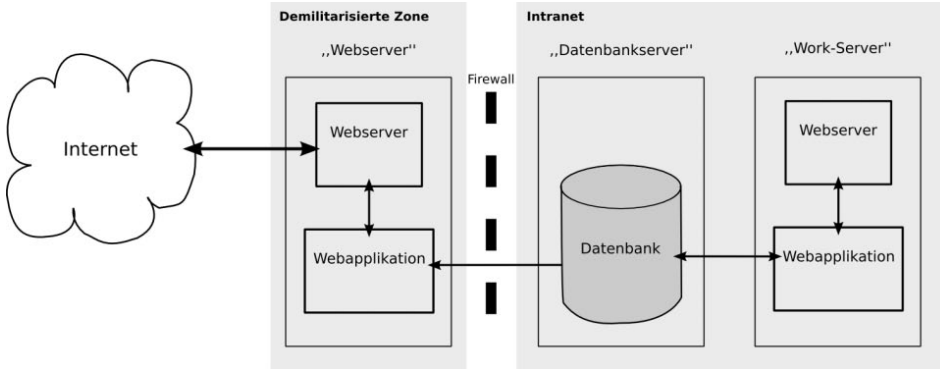


Abbildung 10.3: Schutz des Datenbankservers durch eine Firewall

Datenbankadapter notwendig. Zusätzlich müssen Sie auf dem Datenbankserver den Zugriff über das Netzwerk für die genutzten Kennungen erlauben.

Falls Sie sich Sorgen bezüglich der **Sicherheit** Ihrer Webapplikation machen, können Sie die Pflege der Daten auch von deren Abruf trennen: Sie können z. B. für den Teil der Webapplikation, der das Userinterface für die Eingabe von Daten beinhaltet, eine (Datenbank-)Kennung nutzen, die Schreibrechte auf die verwendete Datenbank besitzt. Der Teil der Webapplikation, der für den Abruf der Daten zuständig ist, nutzt eine Kennung, die nur lesend auf den Datenbestand zugreifen kann. Wenn Sie beide Teile auch noch auf separaten Servern unterbringen und mit Hilfe einer Firewall voneinander trennen können (Abb. 10.3), haben Sie maximalen Schutz bei möglichen Schwachstellen in der Webapplikation.

Weitere Literatur

- »Das XAMPP-Handbuch« von Kai Seidler und Kay Vogelgesang. Addison-Wesley 2006, ISBN 3-8273-2281-2; XAMPP ist eine eigene Distribution, die alle (L)AMP-Teile enthält und speziell auf deren Anwendung ausgerichtet ist.

10.2 MySQL

MySQL kann die Ehre für sich verbuchen, die bekannteste freie Datenbank zu sein, die zusammen mit Webapplikationen wie PHP genannt wird. Viele Content-Management-Systeme auf der Basis von PHP setzen eine MySQL-Installation voraus; daher wollen wir Ihnen hier kurz die Spezifika einer Netzwerkinstallation von MySQL nahebringen.

MySQL	Debian	Fedora	SUSE
Server	mysql-server	mysql-server	mysql
Client	mysql-client	mysql-client	mysql-client
GUI/Webbasierte Administration	phpmyadmin mysql-admin mysql-navigator mysql-query-browser	mysql-administrator	phpMyAdmin mysql-administrator mysql-query-browser
ODBC-Treiber	libmyodbc	mysql-connector-odbc	MyODBC-unixODBC
Headerfiles u.Ä. für die Programmierung	libmysqlclient15-dev	mysql-devel	mysql-devel
Benchmarking und Tests	—	mysql-bench	mysql-bench
PHP-Modul zur Anbindung an MySQL	php4-mysql php5-mysql	php-mysql	php5-mysql

Tabelle 10.1: Paketübersicht

1. Schritt: Den MySQL-Server netzwerkfähig machen

Grundlegende Einstellungen nehmen Sie in der Datei `/etc/mysql/my.cnf` vor. Die für Sie wichtigsten Informationen sind in der Sektion `mysqld` enthalten: der Port und die IP-Nummern, für die sich der Server zuständig fühlt. In der Voreinstellung ist die `bind-address` auf 127.0.0.1 gesetzt, was effektiv einer Sperrung des Netzwerkzugriffs von außen gleichkommt. Wenn der MySQL-Server von außen erreichbar sein soll, müssen Sie ihm die IP-Adresse der Netzwerkkarte als `bind-address` mitgeben. Mehrere IP-Nummern geben Sie, durch ein Leerzeichen getrennt, hintereinander an.

```
[client]
port            = 3306
socket          = /var/run/mysqld/mysqld.sock

[mysqld]
#
# * Basic Settings
#
```

```
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
bind-address    = 192.168.1.10 127.0.0.1
```

Alle anderen Informationen können in der Voreinstellung belassen werden. Beachten Sie, dass die Datei `my.cnf` sowohl vom Server als auch von den Clients ausgewertet wird! Je nach Distribution unterscheiden sich die Informationen darüber, wo z. B. der Socket im Dateisystem liegt und die Prozess-ID des laufenden Server-Prozesses gespeichert wird. Ändern Sie den Speicherort z. B. der Socket-Datei, sollten Sie die Information sowohl in der `mysqld`- als auch in der `client` Sektion ändern. Ansonsten müssten Sie beim Start des Clients diese Information jedes Mal über eine Kommandozeilenoption mitgeben.

2. Schritt: Steuerung des Zugriffs: Alle weiteren administrativen Aufgaben können Sie mit dem MySQL-Client erledigen. Die Zugriffssteuerung erfolgt auf der Basis einer eigenen Nutzerverwaltung. Um in einigen Fällen den Zugriff zu vereinfachen, wird mit Hilfe des Ident-Mechanismus die aktuelle Systemkennung, unter der Sie angemeldet sind, auf eine etwaig vorhandene Datenbankkennung gemappt. Zu beachten ist hier, dass in der Voreinstellung die Kennung `root` (und nur sie) Vollzugriff auf die Datenbank `mysql` besitzt. Dies ist die Systemdatenbank von MySQL, in der die Zugriffsberechtigungen für alle Datenbanken gespeichert werden.

Tipp

Voreingestellt ist ein passwortloses Ident-Login; das bedeutet, dass Sie ohne Passwort die zentrale Systemdatenbank `mysql` bearbeiten können, wenn Sie auf dem gleichen Server unter `root` angemeldet sind. Sollten Sie sich aus irgendwelchen Gründen einmal vom Zugang zur Datenbank `mysql` ausschließen, können Sie den Serverprozess mit der Option `--skip-grant-tables` aufrufen. Damit wird die Rechteverwaltung ausser Kraft gesetzt und Sie können die Zugriffsrechte berichtigen oder ein etwaiges Passwort für die (Datenbank-)Kennung `root` neu setzen.

Wir wollen Ihnen die Kontrolle des Zugriffs auf einen MySQL-Server über das Netzwerk kurz anhand eines gebräuchlichen Bespiels demonstrieren: Wir legen eine Kennung und eine Datenbank gleichen Namens an, die später für eine Webapplikation genutzt werden soll. Interessanterweise geben Sie das Passwort für den Zugriff bei der Freigabe des Zugriffs an. Das bedeutet, dass der Zugriff von verschiedenen Rechnern aus bei gleicher Kennung mit unterschiedlichem Passwort erfolgen kann!

```
hugo:~# mysql mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8 to server version: x.x.xx-xxxx-xxxx

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database beispiel;
Query OK, 0 rows affected (0.00 sec)

mysql> create user beispiel;
Query OK, 0 rows affected (0.00 sec)

mysql> grant all on beispiel.* to 'beispiel'@'theodor' identified by password 'geheim!';
Query OK, 0 rows affected (0.00 sec)

mysql>
```

Jetzt kann vom Rechner *theodor* aus unter Angabe der Kennung `beispiel` (`-u beispiel`) auf die Datenbank `beispiel` auf dem Server `hugo` (`-h hugo`) zugegriffen werden:

```
marcus@theodor:~$ mysql -h hugo -u beispiel beispiel
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 72 to server version: x.x.xx-xxxx-xxxx

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show tables;
[...]
```

Jetzt können Sie auf der Kommandozeile mit Hilfe von SQL-Befehlen Tabellen anlegen etc. Es existiert eine Reihe von grafischen bzw. webbasierten Tools – das bekannteste Webbasierte Tool dürfte phpMyAdmin sein –, die Sie bei der Pflege einer MySQL-Datenbank unterstützen. Aus unserer Erfahrung würden wir Ihnen jedoch raten, grundlegende Datenbankstrukturen immer mit Hilfe von in einer Textdatei gespeicherten SQL-Befehlen anzulegen.

Zum Schluss noch ein kurzer Ausflug nach PHP. In PHP würden Sie mit folgenden Kommandos die Datenbank öffnen:

```
<?php
$dbhandle = mysql_connect("hugo", "beispiel", "geheim!");
$datenbank = mysql_select_db("beispiel", $dbhandle);
?>
```

Die gute englische Dokumentation gibt es auch in einer deutschen Version. Außerdem finden Sie auf der MySQL-Homepage eine Liste grafischer Clients, mit denen Sie die Datenbank administrieren können. Die Syntax für die PHP-Funktionsaufrufe finden Sie auf der PHP-Webseite mit ausführlichen Beispielen.

Weitere Literatur

- »Das offizielle MySQL 5-Handbuch«, Konfiguration, Administration, Entwicklung & Optimierung, ISBN 978-3-8273-2404-7
- »MySQL Online Referenzhandbuch«: `http://dev.mysql.com/doc/refman/5.1/de/`
- »MySQL 5.1 Referenzhandbuch«, Kapitel 5.8 »Allgemeine Sicherheitsaspekte und das MySQL-Zugriffsberechtigungssystem«
- »MySQL 5.1 Referenzhandbuch«, Kapitel 5.9 »MySQL-Benutzerkontenverwaltung«
- »PHP Online-Dokumentation zur MySQL-Unterstützung«: `http://www.php.net/manual/de/ref.mysql.php`

10.3 PostgreSQL

Auch wenn PostgreSQL (noch) nicht den Bekanntheitsgrad von MySQL besitzt, ist es doch ein Datenbanksystem, das seit langem Produktionsreife besitzt. Datenbanken gehören wie Dateisysteme zu den Tools, bei denen Fehlfunktionen schwerwiegende Auswirkungen haben. Daher werden zurzeit vier Versionen aktiv gepflegt: der 8.1er-Zweig stellt die modernste PostgreSQL-Variante dar, die die neuesten Features enthält, gefolgt von der 8.0er-Variante. Zusätzlich werden noch zwei 7er-Varianten mit Sicherheitsupdates versorgt, damit der Nutzer je nach Einsatzzweck und Feature-Wunsch zwischen »superstabil« und »Funktionsreichtum« wählen kann. Unsere Erfahrung zeigt jedoch, dass alle Versionen dem Anspruch »superstabil« genügen und für den produktiven Einsatz geeignet sind.

PostgreSQL	Debian	Fedora	SUSE
Server	postgresql-8.1	postgresql-server	postgresql-server
Client	postgresql-client-8.1	postgresql	postgresql
GUI/Webbasierte Administration	pgadmin3 pgaccess phppgadmin	pgadmin3	pgaccess
Dokumentation	postgresql-doc-8.1	postgresqkl-docs	postgresql-docs
Zusätzliche Funktionen	postgresql-contrib-8.1	postgresql-contrib	postgresql-contrib
Headerfiles u.Ä. für die Programmierung	postgresql-server-dev-8.1	postgresql-devel	postgresql-devel
Benchmarking	in postgresql-contrib	postgresql-test	—

Tabelle 10.2: Paketübersicht

PostgreSQL	Debian	Fedora	SUSE
Einbettbare Programmiersprachen	postgresql-plperl-8.1 postgresql-plpython-8.1 postgresql-pltcl-8.1 postgresql-8.1-pljava-gcj postgresql-8.1-plr postgresql-8.1-plruby	postgresql-python postgresql-pl postgresql-tcl	postgresql-pl (beinhaltet tcl perl und python)
Replikation	postgresql-8.1-slony1	—	—
JDBC/ODBC-Treiber	psqlodbc libpg-java	postgresql-odbc postgresql-.jdbc	psqlODBC postgresql-jdbc
PHP-Modul zur Anbindung an PostgreSQL	php4-pgsql php5-pgsql	php-pgsql	php5-pgsql

Tabelle 10.2: Paketübersicht (Fortsetzung)

Die Konfiguration von PostgreSQL erfolgt über drei Textdateien, die Sie in dem Verzeichnis finden, in dem auch die eigentlichen Datenbanken gespeichert werden, beispielsweise `/var/lib/pgsql/data` oder auch `/var/lib/postgresql/...`. Kleiner Tipp: Wenn Sie dort nach der Installation nur ein leeres Verzeichnis vorfinden, muss die Datenbank noch initialisiert werden. Dies geschieht aber i. d. R. beim ersten Aufruf des PostgreSQL-Servers, wenn Sie diesen über das Init-Script `/etc/init.d/postgresql` starten.

Dabei kommt den drei Dateien jeweils eine eigene Bedeutung zu:

- Mit der Datei `postgresql.conf` konfigurieren Sie den Serverprozess. Hier schalten Sie u. a. ein, ob PostgreSQL nur auf Kontakt über den Socket oder auch über TCP/IP reagieren soll.
- In `pg_hba.conf` wird eingetragen, welcher Nutzer von wo aus wie auf eine Datenbank zugreifen kann. Davon ist jedoch nicht die Vergabe von Rechten innerhalb der Datenbank auf verschiedene Tabellen betroffen. Dies geschieht mit Hilfe des SQL-Kommandos `GRANT`.
- PostgreSQL bietet die Möglichkeit, beim Login eine andere Datenbankkennungen zu benutzen, als dieselbe wie die Systemkennung des Nutzers. Diese Informationen werden in der Datei `pg_ident.conf` abgelegt.

Debian-Tipp

Damit auf einem Debian-System mehrere Versionen parallel gefahren werden können, wird in `/etc/postgresql` für jedes große Release ein Unterverzeichnis angelegt. Darin finden Sie die im Folgenden beschriebenen Konfigurationsdateien. Der Datenbereich wird versionsspezifisch aufgeteilt im Verzeichnis `/var/lib/postgresql/<Version>` angelegt.

Wir wollen Ihnen im Folgenden anhand eines netzbasierten Szenarios die Funktionen der drei Konfigurationsdateien kurz vorstellen. Bei der Nutzung von Post-

greSQL hat sich folgende Vorgehensweise bewährt: Die Kennung `postgres` hat in allen Datenbanken so etwas wie `root`-Rechte. Wir werden die Datenbank so konfigurieren, dass man sie zwar ohne Passwort nutzen kann, aber nur, wenn man auf dem System als Nutzer `postgres` angemeldet ist. Der Zugriff auf die Systemkennung `postgres` kann dann entweder über `root` mit Hilfe von `su` oder über SSH-Schlüssel erfolgen. Alle anderen Datenbankkennungen müssen sich mit einem Passwort autorisieren.

1. Schritt: PostgreSQL »netzwerkfähig« machen: Bei allen Distributionen ist aus Sicherheitsgründen der Zugriff über das Netzwerk ausgeschaltet. Damit der Server auch auf Netzwerkzugriffe reagiert, muss die Option `port` aktiviert und mit einer Portnummer versehen werden. Standard ist für PostgreSQL die Nummer `5432`. Zusätzlich müssen Sie mit dem Schlüsselwort `listen_address` die Netzwerkschnittstellen benennen, auf denen der Server lauschen soll. Diese werden anhand ihrer IP-Nummern bzw. der Hostnamen identifiziert.

Listing 10.1: postgres.conf

```
#---------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#---------------------------------------------------------------------------

# - Connection Settings -
#listen_addresses = 'localhost'         # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost', '*' = all
listen_addresses = localhost, 192.168.1.10
port = 5432

# - Security & Authentication -
ssl = true
```

Wollen Sie den direkten Zugriff auf die Datenbank auch über ungesicherte Netzwerke erlauben, sollten Sie mit der Option `ssl = on` die SSL-Verschlüsselung einschalten. Damit diese funktioniert, benötigen Sie noch ein gültiges Zertifikat, das Sie im Datenverzeichnis von PostgreSQL unter dem Namen `server.crt` bzw. `server.key` ablegen bzw. verlinken müssen. Änderungen an dieser Konfigurationsdatei teilen Sie dem bereits laufenden Server mit Hilfe des Kommandos `pg_ctl reload` mit.

2. Schritt: Wer darf wie auf welche Datenbank zugreifen: Die Zugriffssteuerung erfolgt auf zwei Ebenen: mit der Konfigurationsdatei `pg_hba.conf` bestimmen Sie, welche Kennung überhaupt eine Datenbank öffnen darf. Innerhalb des Systems können Sie dann mit `GRANT`-Kommandos den Zugriff auf Tabellen oder Teile davon steuern.

Die Datei `pg_hba.conf` enthält Voreinstellungen, die wir jetzt für unsere Zwecke abwandeln: Jede Zeile stellt eine Regel dar, die mit sechs verschiedenen Parametern definiert werden kann. In unserem Szenario wollten wir der lokalen System-Kennung `postgres` erlauben, ohne die Eingabe eines Passworts die Datenbankkennung

postgres zu nutzen. Dies wird durch die erste Zeile local all-Zeile eingestellt. Dort ist in der Spalte USER die Datenbankkennung postgres eingetragen.

Die Methode ident sameuser bestimmt, dass System- und Datenbankkennung übereinstimmen müssen. Jedes Mal, wenn Sie eine Datenbankverbindung öffnen, können Sie eine (Datenbank-)Kennung angeben, mit deren Rechten Sie dann die Datenbank nutzen wollen. Mit dem Kommandozeilentool psql könnten Sie sich über die Option -U postgres als (Datenbank-)Nutzer postgres ausgeben und hätten dann in unserem Szenario vollen Zugriff auf alle Datenbanken. Dies wird durch die Methode ident sameuser verhindert.

Das Schlüsselwort local bestimmt, dass dieser Konfigurationseintrag für Zugriffe über den lokalen Socket auf die Datenbank gilt. Da diese Zugriffe über eine Datei im lokalen Dateisystem ablaufen, ist dort kein Eintrag bezüglich IP-Nummer oder Netzmaske zu machen. Den Zugriff über die Netzwerkschnittstelle konfigurieren Sie mit Einträgen des Typs host. Wir haben in unserer Konfiguration drei Einträge dieses Typs. Beginnen wir mit der Erläuterung des letzten Eintrags: Da die Einträge der Reihe nach ausgewertet und auf Zugriffsversuche angewendet werden, können wir mit der letzten Zeile mit der Methode reject eine generelle Ablehnung definieren. Jetzt können wir mit Einträgen, die wir **vor** dieser Zeile einfügen, differenziert den Zugriff erlauben. Der erste Eintrag erlaubt z. B. über die lokale Loopback-Schnittstelle allen (Datenbank-)Kennungen den Zugriff auf alle Datenbanken nach Eingabe des Passwortes (dies wird durch die Angabe der Methode md5 erreicht).

Listing 10.2: pg_hba.conf

```
# TYPE  DATABASE    USER        IP-ADDRESS          IP-MASK             METHOD
# Database administrative login by UNIX sockets
local   all         postgres                                            ident sameuser
# All other connections by UNIX sockets
local   all         all                                                 md5
# All IPv4 connections from localhost
host    all         all         127.0.0.1           255.255.255.255     md5
# Alle Verbindungen aus dem lokalen Subnetz
host    telefon     telefon     192.168.1.0         255.255.255.0       md5
# reject all other connection attempts
host    all         all         0.0.0.0             0.0.0.0             reject
```

Die zweite host-Zeile erlaubt im Subnetz 192.168.1.0 den Zugriff auf eine Datenbank mit dem Namen telefon über die Datenbankkennung telefon. Mit Hilfe des Kommandozeilentools psql, das auf einem Rechner im freigegebenen Subnetz installiert ist, sähe das so aus:

```
victor@theodor:~$ psql -h 192.168.1.10 -U telefon telefon
Passwort für Benutzer telefon:
Dies ist psql 8.1.n, das interaktive PostgreSQL-Terminal.
```

```
Geben Sie ein:  \copyright für Urheberrechtsinformationen
                \h für Hilfe ber SQL-Anweisungen
                \? für Hilfe ber interne Anweisungen
                \g oder Semikolon, um eine Anfrage auszuführen
                \q um zu beenden

telefon=>\dt
# Ausgabe der Tabellen der Datenbank telefon
```

Zu PostgreSQL existiert eine gute, englischsprachige Dokumentation, in der Sie zu allen hier behandelten Themen ausführliche Informationen finden. Diese können Sie entweder als Paket lokal installieren oder über die Homepage von PostgreSQL `www.postgresql.org` abrufen. Im Einzelnen empfehlen wir Ihnen folgende Kapitel:

Weitere Literatur

- »III Server Administration«, »Chapter 16.7. Secure TCP/IP Connections with SSL« für die Einrichtung von SSL-verschlüsselten Verbindungen
- »III Server Administration«, »Chapter 20. Client Authentication« für die Konfiguration des Zugriffs auf den Server. Dort finden Sie auch eine Referenz aller in der Datei `pg_hba.conf` nutzbaren Schlüsselwörter.
- »PostgreSQL-FAQ«: Eine Liste mit häufig gestellten Fragen existiert auch auf Deutsch.

10.4 Zugriff per ODBC

Da sich die SQL-Dialekte verschiedener Datenbanken i. d. R. mehr oder weniger unterscheiden, wurde mit ODBC der Versuch gemacht, einen Standard für den Zugriff auf im Prinzip beliebige Datenquellen (meistens SQL-Datenbanken, vom Prinzip her könnte aber jede Datenquelle genutzt werden) zu entwickeln, für die ein ODBC-Treiber existiert.

ODBC	Debian	Fedora	SUSE
unixODBC			
Basis-Bibliotheken für den Zugriff von Unix auf ODBC-Server	unixodbc unixodbc-bin	unixODBC unixODBC-kde	unixODBC unixODBC-gui-*
FreeTDS			
Freie TDS-Implementation für den ODBC-Zugriff auf MS SQL- und Sybase-SQL-Server	libct3 tdsodbc	—	—

Tabelle 10.3: Paketübersicht

Leider funktioniert die Standardisierung genau andersherum, als man es vermuten würde: Es wird nicht ein ODBC-Wrapper vor die Datenbank geschaltet, die dann mit einem Standard-ODBC-Client genutzt werden kann, sondern das ODBC-Tool

des Betriebssystems muss um Treiberkomponenten für die jeweilige Datenbank ergänzt werden.

Von Windows zu Linux: Wenn Sie unter Windows auf eine MySQL- oder PostgreSQL-Datenbank zugreifen wollen, müssen Sie dort die jeweiligen ODBC-Treiber für MySQL respektive PostgreSQL installieren. Für beide Datenbanken existieren freie Bibliotheken, die sich problemlos installieren und dann über die Windows-eigene ODBC-Konfiguration einbinden lassen.

Von Linux zu Windows: Andersherum sieht es schon etwas schwieriger aus: Wenn Sie von Linux aus mit Hilfe von ODBC auf andere SQL-Datenbanken zugreifen wollen, benötigen Sie die passende Bibliothek. Für Linux existiert mit FreeTDS eine ODBC-Bibliothek, die über das TDS-Protokoll Zugriff auf MS-SQL-Server und Sybase-SQL-Server erlaubt. Diese Bibliothek nutzt das UnixODBC-Projekt, das den Rahmen für ODBC-Zugriffe auf andere Datenquellen zur Verfügung stellt. Dort finden Sie auch einen Überblick über zum Teil proprietäre Erweiterungen z. B. für den Zugriff auf Oracle-Datenbanken.

Der Zugriff auf MySQL- bzw. PostgreSQL-Datenbanken aus PHP oder anderen Programmiersprachen sollte aber i. d. R. mit Hilfe der nativen Clients und Bibliotheken erfolgen, die für alle Betriebssysteme existieren, und der bevorzugte Weg sein. Nur mit diesen Clients und Bibliotheken können Sie die Spezialfunktionen nutzen, die das jeweilige Datenbanksystem bietet.

Weitere Literatur

- »UnixODBC«: `http://www.unixodbc.org/`
- »FreeTDS«: `http://www.freetds.org/`
- »MySQL-ODBC-Dokumentation«: Die MySQL-Dokumentation enthält eine ausführliche Beschreibung, wie Sie von Windows aus per ODBC auf MySQL-Datenbanke zugreifen können. `http://dev.mysql.com/doc/refman/5.0/en/myodbc-connector.html`

11 Sicherheit im Netzwerk

Wir haben in diesem Buch schon mehrfach die Vorteile der Secure Shell (SSH) gepriesen, die mit geringem Aufwand eine gut abgesicherte Kommunikation ermöglicht, auch über Netzwerkverbindungen hinweg, die in böser Absicht manipuliert oder abgehört werden könnten.

11.1 Sicher mit der Secure Shell arbeiten

Die Einrichtung einer SSH-Verbindung beschränkt sich zunächst auf die Installation des OpenSSH-Clients auf der einen und des zugehörigen Server-Dämons auf der anderen Seite (für beide hält jede gängige Distribution fertige Pakete bereit). Die ausgelieferten Konfigurationsdateien sehen meist vor, dass zum Nachweis, dass der Benutzer auf dem Zielrechner Programme bzw. eine Shell starten darf, die Kenntnis einer Benutzerkennung und des zugehörigen Passworts genügen.

Das Passwort wird von der SSH nicht im Klartext übertragen und auch nicht dem Server in der Originalform bekannt gemacht, sondern es wird nur auf sichere Weise geprüft, ob das vom Benutzer eingegebene Passwort mit dem auf der Serverseite verschlüsselt hinterlegten Wert übereinstimmt.

Die Authentisierung per Passwort darf also durchaus als sicher gelten; allerdings besteht das Risiko, dass ein Angreifer über längere Zeit hinweg eine riesige Anzahl von Passwörtern durchprobieren und dabei das passende herausfinden könnte. Aber bei häufiger Benutzung der SSH wird das ständige Eintippen des Passworts bald lästig. Keinesfalls sollte nun als Abhilfe das Passwort verkürzt oder ganz darauf verzichtet werden!

11.1.1 Anmelden ohne Passwort: SSH-Schlüssel

Es gibt eine bessere Lösung: Man kann einen kryptografisch sicheren (das hoffen die Experten jedenfalls) Schlüssel generieren lassen, ihn – beispielsweise über eine per Passwort authentisierte SSH-Verbindung – zum Serverrechner übertragen und dort als Zugangsberechtigung zu einer Kennung hinterlegen. Um diese Methode noch sicherer zu machen, besteht der Schlüssel zudem aus zwei Teilen: einem öffentlichen Teil (»public key«), mit dessen Hilfe serverseitig die Authentizität der Anmeldung überprüft wird, und einem geheimen Teil (»secret key«), der clientseitig benutzt wird und nur dem berechtigten Nutzer zugänglich sein darf.

Zur Generierung eines Schlüssels dient das Kommando `ssh-keygen`, für das übrigens – wie auch zur Installation des Schlüssels – keine root-Rechte erforderlich sind:

```
victor@theodor:~$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/victor/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):ggehheim
Enter same passphrase again:ggehheim
Your identification has been saved in /home/victor/.ssh/id_dsa.
Your public key has been saved in /home/victor/.ssh/id_dsa.pub.
The key fingerprint is:
98:fd:8f:fc:8a:4a:8e:6c:4e:65:34:d8:c1:13:09:c5 victor@theodor
victor@theodor:~$ ssh hugo "mkdir -p -m 700 .ssh"
Password:log+pass
victor@theodor:~$ cat .ssh/id_dsa.pub | ssh hugo "cat >>.ssh/authorized_keys"
Password:log+pass
victor@theodor:~$ ssh hugo id
Enter passphrase for key '/home/victor/.ssh/id_dsa':ggehheim
uid=1004(victor) gid=1004(victor) Gruppen=1004(victor)
victor@theodor:~$
```

Die Option `-t dsa` legt den Schlüsseltyp fest[1]; der geheime Teil des Schlüssels wird standardmäßig in der Datei `id_dsa` abgelegt und der öffentliche in `id_dsa.pub` im Unterverzeichnis `.ssh` des Home-Verzeichnisses.

Mit den nachfolgenden Kommandos wird zunächst über klassische SSH-Aufrufe mit Angabe des Login-Passworts ein `.ssh`-Verzeichnis auf dem anderen Rechner angelegt und sodann der öffentliche Schlüssel dort in die Datei `authorized_keys` kopiert bzw. an diese angefügt, sofern sie schon existiert. Durch die Eintragung in diese Datei wird der Schlüssel berechtigt, die Kennung zu benutzen, der das zugehörige Home-Verzeichnis gehört.

Den Effekt sehen Sie anhand des letzten Kommandos: Jetzt wird nicht mehr nach dem Login-Passwort gefragt, sondern stattdessen nach dem Passwort, mit dem der Schlüssel gesichert wurde.

Aber schon wieder muss ein Passwort eingetippt werden! Dieses soll nun verhindern, dass jemand, der den geheimen Schlüssel irgendwo (z. B. auf einer alten Festplatte oder im Backup) findet, dadurch Zugang zum Zielrechner erhält. Aber was nützt der schöne Schlüssel als Passwortersatz, wenn man ihn bei jeder Benutzung wieder durch ein Passwort entsperren muss?

Zwar kann man bei der Schlüsselerzeugung ein leeres Passwort vergeben; dann ist nachher ein direkter Zugriff auf den Zielrechner ohne jedes Passworttippen möglich. Aber damit entfällt auch der Schutz des Schlüssels.

[1] Alternativ kann auch der Schlüsseltyp `rsa` benutzt werden. Nach heutigem Erkenntnisstand können beide Typen als hinreichend sicher gelten.

11.1.2 Ein Agent besorgt die Schlüssel

Eine bessere Lösung besteht in der Verwendung des SSH-Agenten. Er verwaltet in seinem Arbeitsspeicher (nicht auf der Festplatte!) einen oder mehrere SSH-Schlüssel und stellt diese bei Bedarf zur Authentisierung von SSH-Verbindungen bereit. Hierzu legt er unter `/tmp` einen für andere Benutzer nicht les- oder schreibbaren Socket an, dessen Name über Shellvariablen an alle weiteren SSH-Aufrufe übermittelt wird. Diese setzen sich mit dem Agenten in Verbindung und erhalten damit Zugriff auf den entsperrten Schlüssel. Auf die gleiche Art kann das Kommando `ssh-add` dem Agenten einen Schlüssel zur Aufbewahrung übergeben, zu dem es vom Benutzer das Passwort erfragt hat:

```
victor@hugo:~$ ssh-add
Enter passphrase for /home/victor/.ssh/id_dsa:ggehheim
Identity added: /home/victor/.ssh/id_dsa (/home/victor/.ssh/id_dsa)
victor@hugo:~$
```

Normalerweise schränkt die Weitergabe von Informationen über Shellvariablen die Nutzbarkeit des Agenten auf lästige Weise ein, da die Variableninhalte von der Shell immer nur an die eigenen Kindprozesse übergeben werden. Die grafischen Oberflächen wie z. B. KDE sind jedoch meist so vorkonfiguriert, dass sie beim Start einer Sitzung den SSH-Agenten starten und dass die Schlüsselinformationen an sämtliche Prozesse weitergegeben werden, die innerhalb der Sitzung gestartet werden. Es genügt dann also, zu Beginn der Sitzung einmal mit `ssh-add` den Schlüssel zu entsperren, und von da an kann man sich über die Sicherheit *und* Bequemlichkeit freuen, die die SSH bietet.

11.1.3 SSH (noch) sicherer machen

Trotz der sehr weitgehenden Sicherheit, die die SSH bietet, bleiben noch gewisse Restrisiken. Beispielsweise wird der SSH-Port (22) eines jeden mit dem Internet verbundenen Rechners von Hackern aus aller Welt unermüdlich mit Login-Versuchen bombardiert, um durch Ausprobieren gängiger Kennungen und Passwörter eine gültige Kombination und damit Zugang zu dem System zu finden. Viele Administratoren schwören daher auf ergänzende Maßnahmen, durch die sie hoffen, ihr System noch sicherer zu machen und einen Missbrauch des Secure-Shell-Zugangs auszuschließen, der ja meist zu einem voll funktionsfähigen Shell-Prompt führt und deshalb besonders riskant erscheint. Hier sind einige der »Patentrezepte« und unser Kommentar dazu:

- Die veraltete Protokollversion 1 der SSH sollte grundsätzlich abgeschaltet werden, sofern nicht unbedingt uralte Clients unterstützt werden müssen. Dies erreicht man durch den Eintrag `Protocol 2` in der Server-Konfigurationsdatei `sshd_config`.
- Anstatt den Standard-Port 22 zu nutzen, kann man den SSH-Dämon auf einem anderen Port auf Verbindungsversuche lauschen lassen. Hierzu ist in der `sshd_config`-Datei eine entsprechende `Port`-Zeile einzutragen; clientseitig kann die

Portnummer im Kommando (durch die Option `-p`) vorgegeben oder ein für den Zielhost spezifischer Abschnitt in die Konfigurationsdatei `/.ssh/config` (am Ende) eingetragen werden, der die gleiche Port-Zuweisung enthält:

```
Host hugo
Port 17022
```

Diese Sicherheitsvorkehrung ist etwas lästig, weil bei jeder Verbindung der Port-Parameter angegeben oder auf jedem Client die Konfiguration ergänzt werden muss. Die Anzahl der Einbruchsversuche (und damit deren Erfolgschance) sinkt, wodurch sich vielleicht der Nachtschlaf des Administrators bessert; einen gezielten Angriff, der systematisch alle Portnummern durchprobiert, kann man dadurch aber nicht abblocken.

- Die Authentisierung über Kennung und Passwort kann generell abgeschaltet werden. Dazu werden die Optionen `PasswordAuthentication` und `ChallengeResponseAuthentication` auf `no` gesetzt. Dies verurteilt das Ausprobieren von Passwörtern zur Erfolglosigkeit – und ist insofern durchaus empfehlenswert –, macht aber auch den Remote-Zugriff unmöglich, wenn kein passender Schlüssel auf der Clientseite verfügbar ist. Deshalb kann diese Maßnahme den Weltreisenden, der sich dringend von unterwegs auf seinem Rechner einloggen will, sehr unglücklich machen.
- Es ist möglich, nur bestimmten, wenigen Benutzern die Anmeldung beim SSH-Dämon zu gestatten. Dazu kann eine Zeile wie z. B.

  ```
  AllowUsers victor
  ```

  in die `sshd_config`-Datei eingetragen werden. Das Ausprobieren von Passwörtern anderer Kennungen ist dann ebenso ausgeschlossen wie eine Gefährdung des Systems durch ein schlecht gewähltes Passwort oder einen verschlampten Schlüssel seitens anderer Benutzer. Ein zugelassener User kann nötigenfalls auf dem Zielsystem mit `su` in eine andere Kennung wechseln und auch die root-Kennung benutzen.
- Verfahren wie das im nächsten Kapitel kurz beschriebene Port Knocking halten zwar hartnäckige, gezielt gegen Ihren Rechner vorgehende Benutzer kaum fern, machen aber die automatisierten Portscans wirkungslos: Nur wer das individuelle Anklopfsignal kennt (das aus einer Folge von Paketen an bestimmte Ports besteht), kann Verbindungen aufnehmen und z. B. das SSH-Login nutzen.
- Durch Firewall-Regeln kann die Nutzung des SSH-Zugangs auf bestimmte Netze oder IP-Adressen beschränkt werden, und es ist in Verbindung mit Erweiterungsmodulen sogar möglich, zu häufige Verbindungsversuche von immer der gleichen IP-Adresse automatisch zu blockieren. Diese Maßnahmen bremsen die typischen Angreifer aus; das Aussperren nach Herkunftsadresse wirkt allerdings nicht, wenn die Angreifer gut organisiert sind und von ständig wechselnden IP-Adressen aus agieren. Im ungünstigen Fall kann es sogar berechtigte Zugriffe behindern.

  Die Einschränkung auf (halbwegs) vertrauenswürdige Adressen oder Adressbereiche hingegen ist sehr sinnvoll, sofern jedenfalls nicht Weltreisenden das Login möglich sein muss.

11.2 Virtuelle private Netzwerke (VPN)

Nehmen wir einmal an, Sie gehen auf Reisen und wollen von unterwegs auf Ihr heimisches (privates oder Firmen-) Netzwerk zugreifen, sei es, um Daten auf Ihren Drucker auszugeben (wobei der Umweg über E-Mail vielleicht zu viel Zeit oder fremde Hilfe braucht) oder um Daten von Ihrem Dateiserver herunterzuladen, die Sie dringend benötigen. Wir könnten Ihnen drei Lösungsmöglichkeiten vorschlagen:

Möglichkeit 1: Sie lassen Zugriffe aus dem Internet in Ihr Heimnetz zu. Durch eine Firewall können Sie diese auf wenige, wirklich benötigte Ports (für den Druckdienst und den Dateizugriff, beispielsweise mittels Samba oder Secure Shell) beschränken, aber dennoch reißen Sie damit ein Sicherheitsloch auf: Auch wenn Sie die Nutzung des Dateiservers durch ein sorgfältig gewähltes Passwort schützen, kann ein geduldiger Hacker – ganz gleichgültig, ob er im Nachbarhaus oder im fernen Osten sitzt – durch wochen- oder monatelanges Durchprobieren verschiedenster Buchstaben- und Sonderzeichenkombinationen irgendwann doch einmal Ihr Passwort erraten und damit auf Ihre Geschäftsgeheimnisse und Steuererklärungen zugreifen.

Außerdem hat es in der Vergangenheit Schwachstellen in den Implementierungen ganz vieler Netzwerkdienste gegeben (dies gilt leider auch für die Secure Shell), die Angreifern eine gewisse Zeit lang bis zum Bekanntwerden und zur Behebung des Fehlers und bis zum Einspielen der Korrekturen freien Zugang zu den Systemen gestatteten. Zwar sind die betroffenen Programme daraufhin verbessert worden und sollen nun auch immun gegen ganze Klassen ähnlicher Attacken sein, doch kann niemand garantieren, dass nicht in irgendeinem Netzwerkdienst neuartige, gravierende Lücken gefunden werden, die eine Umgehung von Passwort- oder anderen Sicherheitsprüfungen ermöglichen.

Deshalb könnten Sie Ihre von außen ansprechbaren Dienste verstecken, indem Sie etwa einen Secure-Shell-Dämon nicht auf dem üblichen TCP-Port 22 auf Verbindungswünsche lauschen lassen, sondern auf einem frei erfundenen und üblicherweise unbenutzten Port. (Nicht dafür geeignet sind diejenigen Portnummern, die für ausgehende Verbindungen verwendet werden; welche das sind, erfahren Sie durch das Kommando `cat /proc/sys/net/ipv4/ip_local_port_range`, das meist den Bereich von 32768 bis 61000 anzeigt.)

Einerseits bewahrt Sie das vor den Nachstellungen all der einfallslosen Hacker, die das ganze Internet nach Secure-Shell-Servern durchsuchen und diese dann mit endlos langen Passwortlisten angreifen; andererseits kann der Angreifer durchaus bei Ihrem Rechner sämtliche Ports von 1 bis 32767 durchprobieren und alle Services entdecken, die über einen davon ansprechbar sind. Sie erreichen also keine wirkliche Sicherheit, sondern verringern nur die Anzahl der zu erwartenden Angriffe (was natürlich auch schon manchen Administrator gerettet hat).

Es gibt noch etwas wirksamere Methoden, die offenen Netzwerkports zu verstecken, wie beispielsweise das »Port Knocking«: Durch Firewall-Regeln weisen Sie

grundsätzlich alle Pakete ab, die z. B. eine neue Verbindung zum ssh-Port eröffnen wollen. Nur wenn eine Reihe von eigentlich unsinnigen Datenpaketen innerhalb kurzer Zeit empfangen werden, die an bestimmte, individuell festgelegte Portnummern gerichtet sind, öffnet ein Serverprozess für kurze Zeit den ssh-Zugang. Der berechtigte Client muss also mit solchen Paketen erst einmal »anklopfen«, ehe er die SSH-Verbindung aufnehmen kann. Über die Wirksamkeit eines solchen Schutzes streiten sich die Gelehrten gern; wenn Sie damit experimentieren wollen, können Sie beispielsweise unter `http://www.zeroflux.org/knock/` eine Implementierung finden, die auch in einigen Distributionen (etwa als Paket `knockd`) enthalten ist.

Neben den Sicherheitsbedenken ergibt sich noch ein weiteres Problem bei der Freigabe von Diensten zum Internet hin: Meist erscheinen interne Netzwerke nach außen hin nur unter einer einzigen IP-Adresse. Wenn Sie also auf gleiche Dienste mehrerer interner Rechner von außen zugreifen wollen, müssen Sie entweder die Dienste auf jeweils unterschiedliche Ports konfigurieren und alle diese Ports in der Firewall freischalten oder eine Umsetzung von Ports zwischen externem und internem Netz einrichten (bei einer Iptables-Firewall mit einer entsprechenden DNAT-Angabe). Beides ist in der Praxis lästig, da Sie clientseitig immer unterschiedliche Portnummern konfigurieren müssen und dies zudem nicht bei jedem Netzwerkdienst problemlos möglich ist.

Betrachten wir deshalb die **Möglichkeit 2**: Sie ziehen ein ziemlich langes, zusätzliches (Cross-over-) Netzwerkkabel zwischen Ihrem Laptop und dem heimischen Netzwerk. Wenn wir für's Erste das Problem, das Kabel bis ins Urlaubsquartier oder den fahrenden Zug zu verlegen, mal außer Acht lassen, ist diese Lösung recht einfach und bequem: Sie bauen in den Laptop und einen der Heimrechner eine zusätzliche Netzwerkkarte ein, geben ihr jeweils eine Adresse in einem privaten Subnetz und lassen den stationären Rechner die Pakete zwischen dem Laptop und dem Heimnetz hin- und herrouten. So kann sich Ihr Laptop mit allen Rechnern und Druckern daheim unterhalten, sofern der als Vermittler dienende Server die Pakete durchlässt – und insoweit die lokalen Firewalls Verbindungen aus dem »mobilen Kabelnetz« akzeptieren.

Bevor wir auf die Unzulänglichkeiten des langen Kabels näher eingehen, lassen Sie uns diese Alternative etwas detaillierter durchspielen:

Angenommen, Ihr Heimnetz benutzt das 24-Bit-Subnetz 192.168.42.0 und die neuen Netzwerkkarten werden auf beiden Seiten als `eth1` erkannt; für das auf dem

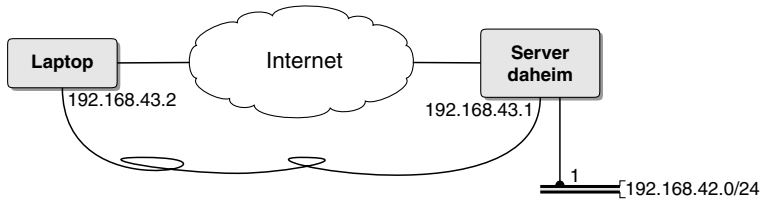


Abbildung 11.1: Laptop an der langen Leitung

Kabel gebildete neue Netzwerk wählen wir die Adresse 192.168.43.0/24. Zwar würde auch ein kleineres Netz z. B. mit einer 30-Bit-Netzmaske ausreichen, das ja auch zwei nutzbare Adressen bietet, aber da wir private, nicht ins Internet geroutete Netzwerke verwenden, brauchen wir mit Adressen nicht zu geizen.

Durch die Kommandos

```
hugo:~# ip link set eth1 up
hugo:~# ip addr add 192.168.43.2/24 dev eth1
hugo:~# ip route add 192.168.42.0/24 via 192.168.43.1
```

auf dem Laptop geben Sie diesem die Adresse 192.168.43.2 für das Interface zur langen Leitung und informieren ihn, dass das heimische Netz über die Gegenstelle mit der Adresse 192.168.43.1 erreichbar ist.

Auf der anderen Seite erhält das Interface eine dazu passende Definition:

```
hugo:~# ip link set eth1 up
hugo:~# ip addr add 192.168.43.1/24 dev eth1
hugo:~# echo 1 >/proc/sys/net/ipv4/ip_forward
```

Natürlich entfällt hier ein gesonderter Routing-Eintrag, da ohnehin außer dem Laptop niemand sonst über diese Schnittstelle zu erreichen ist. Damit zwischen diesem und dem lokalen Netz Pakete ausgetauscht werden können, ist das Forwarding zu aktivieren. (Wenn dieser Rechner, wie in der Zeichnung dargestellt, auch das Gateway zum Internet darstellt, ist das Forwarding ohnehin schon eingeschaltet.) Falls das Routing beschränkt werden soll – der Laptop zum Beispiel nicht mit allen internen Stationen reden darf –, können durch Firewall-Regeln in der FORWARD-Kette entsprechende Festlegungen getroffen werden.

11.2.1 Tunnelbau

Somit hätten wir die Aufgabe der Anbindung des mobilen Rechners perfekt gelöst – bliebe da nicht das Problem, dass uns Ihr transkontinentales Netzwerkkabel womöglich bei nächster Gelegenheit um den Hals gewickelt würde, wenn Sie mit Ihrem Laptop an uns vorbeifahren. Deshalb möchten wir nun als **Möglichkeit 3** vorschlagen, dieses Kabel durch eine virtuelle Verbindung zu ersetzen (das schont Ihren Geldbeutel und unseren Hals).

Dazu benötigen Sie ein Programm, das eine dedizierte physikalische Verbindung simuliert und auf beiden beteiligten Rechnern je ein virtuelles Netzwerkinterface zur Verfügung stellt, das die Datenpakete genau so entgegennimmt wie eine echte Netzwerkkarte, um sie an das Programm weiterzuleiten. Das Programm wiederum verpackt die Daten in normale Netzwerkpakete und sendet sie über eine gewöhnliche IP-Verbindung an die Gegenseite, wo sie ausgepackt und über das dortige virtuelle Interface an den Linux-Netzwerkstack übergeben werden – so, als seien sie über eine direkte Kabelverbindung angekommen. In Analogie zur Modell-

eisenbahn, bei der man sich auch immer fragt, wieso ein Zug, der am einen Ende der Anlage in einen Berg hineinfährt, kurze Zeit später am anderen Ende wieder aus einem anderen Berg herauskommt, spricht man hier von einer Tunnelverbindung.

Zum Tunneln der Datenpakete eignet sich eigentlich das UDP-Protokoll am besten. Es garantiert zwar nicht, dass die Daten vollständig beim Empfänger ankommen, aber das tut eine reale Verbindung auch nicht. Beispielsweise gehen im Wireless LAN bei mäßigem Empfang hin und wieder Pakete verloren, und die höheren Protokollschichten wissen in der Regel mit diesen Verlusten umzugehen. So sorgt das TCP-Protokoll nötigenfalls für eine wiederholte Übertragung verlorengegangener oder »beschädigter« Pakete.

Wird TCP zum Tunneln einer Verbindung benutzt, innerhalb derer wiederum TCP zum Einsatz kommt, kann es passieren, dass bei Störungen der Übertragung auf beiden Ebenen an Fehlern herumkorrigiert wird, was zumindest ineffizient wird. Daher ist TCP/IP für das Tunnelbauen nicht die erste Wahl, funktioniert aber dennoch im Regelfall zufriedenstellend.

Eine ganze Reihe weiterer IP-basierter Protokolle ist speziell für den Transport getunnelter Datenpakete entwickelt worden; hierzu zählen insbesondere die vom IPv6-Standard abgeleiteten AH (Authentication Header) und ESP (Encapsulated Security Payload), die für die Integritätssicherung der Daten bzw. deren Verschlüsselung auf dem Transportweg optimiert sind, allerdings bisweilen Probleme bereiten, weil manche Firewalls sie nicht passieren lassen.

11.2.2 Virtuelle und zugleich private Netze

Wenn mehr als zwei Partner über Tunnelverbindungen miteinander kommunizieren oder die Verbindung nicht nur für ein bestimmtes Protokoll (eine Portnummer) eingerichtet ist, sondern – wie ein richtiges Kabel – Daten beliebiger Protokolle transportieren kann, spricht man von einem virtuellen Netz.

Da die getunnelten Daten über ganz normale Internet-Verbindungen geleitet werden, kann man bei einem virtuellen Netz nie ausschließen, dass ein Fremder den Datenverkehr mitliest oder sogar durch seinen Computer leitet und in böser Absicht manipuliert. Auch bei einem realen, kabelgebundenen Netzwerk ist dies grundsätzlich möglich, erfordert jedoch physischen Zugang und birgt somit ein größeres Risiko, entdeckt zu werden: Würde jemand in unserem obigen Beispiel das »fliegende« Kabel zwischen Laptop und festem Netzwerk anzapfen, so könnte man ihn leicht ausfindig machen; wenn Sie aber Ihre Daten beispielsweise über einen drahtlosen Internet-Zugang tunneln, kann jeder sie unbemerkt abgreifen, der sich im Sendebereich desselben WLANs aufhält.

Deshalb verschlüsselt man die Daten sinnvollerweise auf dem Übertragungsweg, und zwar mit Hilfe des Programms, das an den Tunnelenden die Pakete ein- bzw. auspackt. Erst dann kann man mit Recht von einem VPN, einem virtuellen *privaten* Netz, sprechen.

Im Paketangebot Ihrer Linux-Distribution werden Sie viele Programme finden, mit deren Hilfe Sie ein VPN oder verschlüsselte Tunnelverbindungen realisieren können. Wir wollen hier lediglich drei herausgreifen und etwas näher beschreiben. Das bedeutet nicht, dass die anderen schlecht oder unbrauchbar seien[2]; die im Folgenden genannten können aber als sehr verlässlich angesehen werden und sind einfach anzuwenden.

11.2.3 Secure-Shell-VPN

Die Secure Shell (SSH) erfreut sich bei Linux- (und Unix-) Administratoren großer Beliebtheit, und viele nutzen sie vertrauensvoll zur Fernwartung von Servern und zum Kopieren von Daten über das Netz (per SCP), wie wir in Abschnitt 3.2 auf S. 61 schon ausführlicher beschrieben haben. Insofern bietet es sich geradezu an, sie auch zum Aufbau verschlüsselter Tunnels und damit zur Konstruktion eines VPN zu nutzen. Aber erst die neuesten Versionen (ab 4.3) der OpenSSH sind imstande, Tunnelverbindungen herzustellen, über die beliebige andere Protokolle geleitet werden können. Wir beschreiben dieses Feature, obwohl es derzeit noch als experimentell gilt, weil es unserer Erfahrung nach schon zuverlässig funktioniert und sehr einfach zwischen Rechnern eingesetzt werden kann, die ohnehin schon per SSH miteinander kommunizieren.

Dazu müssen Sie zunächst auf der entfernten Seite des zu grabenden Tunnels – natürlich mit root-Rechten – die Zeile

```
PermitTunnel point-to-point
```

in die `sshd_config`-Datei eintragen (die z. B. im Verzeichnis `/etc/ssh` zu finden sein kann) bzw. eine existierende Zeile entsprechend abändern. Sodann bauen Sie eine neue SSH-Verbindung auf und verlangen mit der zusätzlichen Option `-w` die Einrichtung eines Tunnels. Hierfür muss wiederum die root-Kennung auf der entfernten Seite in Anspruch genommen werden. Andernfalls erscheint beim Login die Meldung `administratively prohibited: open failed`, und das Login wird zwar akzeptiert, aber der Tunnel wird nicht eingerichtet.

Als Adressen für das virtuelle private Netz wählen wir wieder die 192.168.43.1 und 192.168.43.2, und die SSH-Verbindung lässt sich unmittelbar nutzen, um auf der anderen Seite das VPN-Interface zu konfigurieren:

```
theodor:~# ssh root@hugo -w any:any
Password:
Last login: Mon Jul 17 21:18:01 2006 from theodor.beispiel.org
Linux hugo 2.6.16-1-686-smp #1 SMP Tue Sep 27 13:10:31 JST 2006 i686

The programs included with the Debian GNU/Linux system are free software;
```

[2] Einige freilich gelten als unsicher; informieren Sie sich im Zweifelsfall im Internet über den Stand der Erkenntnisse, bevor Sie sich mit der CIA oder der Mafia anlegen …

```
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
hugo:~# ip link show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0d:88:3a:55:6b brd ff:ff:ff:ff:ff:ff
3: tun0: <POINTOPOINT,MULTICAST,NOARP> mtu 1500 qdisc noop qlen 500
    link/[65534]
hugo:~# ip link set tun0 up
hugo:~# ip addr add 192.168.43.1 peer 192.168.43.2 dev tun0
```

Ein paar Erklärungen sind wir Ihnen hierzu noch schuldig:

- Die Option `-w any:any` besagt, dass die Nummern der einzurichtenden virtuellen Schnittstellen auf beiden Seiten beliebig sind und von der SSH gewählt werden sollen. Das Kommando `ip link show` zeigt hier, dass das Interface auf der Seite des entfernten Rechners (`hugo`) `tun0` heißt.
- Mit `ip link set tun0 up` wird dieses Interface aktiviert, also gewissermaßen das Link-Signal auf die Leitung gelegt.
- Bei der Adresszuweisung an `tun0` tauchen hier statt einer Netzmaske das Schlüsselwort `peer` und eine zweite Adresse auf, weil die SSH eine Punkt-zu-Punkt-Verbindung eingerichtet hat (siehe das Attribut POINTOPOINT[3] in der Ausgabe zu `ip link show`). Damit wird das gesamte Link-Layer-Protokoll eingespart, das heißt, die Datenpakete enthalten keine Ethernet-Header, es gibt keine MAC-Adressen, also auch kein ARP, und deswegen muss statisch festgelegt werden, welche IP-Adresse am anderen Ende der Leitung zu finden ist.

Jetzt fehlt noch die Konfiguration des diesseitigen Tunnelendes, die Sie am besten in einem anderen Konsolenfenster auf der Clientseite (`theodor`) – ebenfalls unter der root-Kennung – vornehmen:

```
theodor:~# ip link show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ath0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 200
    link/ether 00:05:4e:4c:2b:30 brd ff:ff:ff:ff:ff:ff
3: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:11:25:44:ab:4d brd ff:ff:ff:ff:ff:ff
4: tun0: <POINTOPOINT,MULTICAST,NOARP> mtu 1500 qdisc noop qlen 500
    link/[65534]
```

[3] Dass hier gern ein »T« eingespart wird, ist genauso sinnig (und fehlerträchtig) wie das gesparte »n« im `u(n)mount`-Kommando.

```
theodor:~# ip link set tun0 up
theodor:~# ip addr add 192.168.43.2 peer 192.168.43.1 dev tun0
theodor:~# ping 192.168.43.1
PING 192.168.43.1 (192.168.43.1) 56(84) bytes of data.
64 bytes from 192.168.43.1: icmp_seq=1 ttl=64 time=1.78 ms
64 bytes from 192.168.43.1: icmp_seq=2 ttl=64 time=1.50 ms
64 bytes from 192.168.43.1: icmp_seq=3 ttl=64 time=1.53 ms

--- 192.168.43.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 1.502/1.607/1.784/0.133 ms
theodor:~# ip route add 192.168.42.0/24 via 192.168.43.1
theodor:~#
```

Hier wird die komplementäre Adressenzuweisung vorgenommen. Damit ist das virtuelle Kabel betriebsbereit, wie man mit `ping` überprüfen kann.

Durch den Routing-Eintrag für das Netz 192.168.42.0/24 kann dieses auch von `theodor` aus erreicht werden, um beispielsweise auf den dortigen Fileserver oder den Drucker zuzugreifen. Sämtliche Protokolle – auch die, die ansonsten als unsicher gelten müssen – können über das VPN geschützt abgewickelt werden. Obwohl die Datenpakete den weiten Weg über das freie, wilde Internet nehmen, sind sie durch das SSH-Protokoll gegen fremden Einblick und gegen Manipulationen geschützt. Auch wenn zur Konfiguration root-Rechte benötigt werden, können selbstverständlich die Daten aller User das VPN nutzen, genau wie bei Verwendung einer realen Kabelverbindung.

Die Einrichtung dieses VPN haben wir hier sozusagen in Handarbeit demonstriert, und genau darin sehen wir das Hauptanwendungsgebiet für das SSH-VPN: Ohne umständliche Vorbereitung kann zwischen zwei Rechnern, die per SSH kommunizieren können und für die man Administratorrechte besitzt, ein sicherer Tunnel ad hoc eingerichtet werden. Die netzwerkseitigen Voraussetzungen beschränken sich darauf, eine TCP/IP-Verbindung zum Port 22 des Zielrechners aufnehmen zu können, was oft auch aus fremden oder öffentlichen Netzen heraus möglich ist[4].

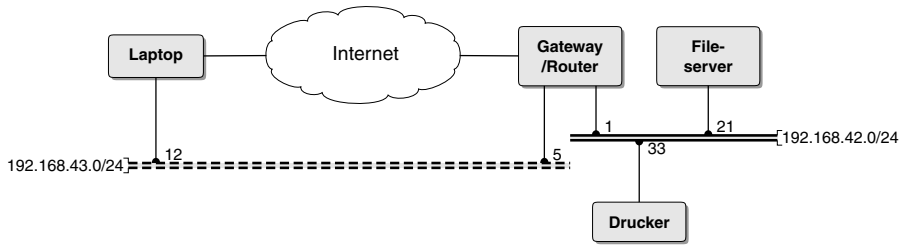


Abbildung 11.2: VPN-Netzstruktur

[4] Wenn Sie einen SSH-Dämon auf Port 80 laufen lassen, können Sie Ihr VPN auch durch restriktive Firewalls hindurchmogeln – sofern die Verbindung nicht über einen HTTP-Proxyserver gezwungen wird.

Einen vollwertigen Ersatz für eine »richtige« VPN-Software bietet die SSH dennoch nicht. Zwar lassen sich die Abläufe ein wenig automatisieren, aber dabei stößt man bald an Grenzen: Tunnel-Devices mit wechselnden Nummern sind schwierig zu konfigurieren, und ein automatisches Wiederaufsetzen nach Unterbrechung der Verbindung ist vonseiten der SSH nicht vorgesehen.

11.2.4 OpenVPN

Neben der Secure Shell, die das VPN-Geschäft erst vor kurzem und sozusagen im Nebenberuf erlernt hat, gibt es zahlreiche spezialisierte VPN-Lösungen, sowohl als kommerzielle wie auch als freie Software. Einige von ihnen benutzen IPSec-(IP-Security-)Protokolle, die eigentlich für IPv6, den Nachfolgestandard der jetzigen IP-Version 4 (kurz IPv4 genannt), konzipiert worden sind und wegen dessen schleppender Einführung in die IPv4-Welt zurückportiert wurden. Als Open-Source-Projekt ist hier insbesondere FreeS/WAN bzw. dessen Nachfolger Openswan zu nennen, dessen Software in die meisten Linux-Distributionen Eingang gefunden hat. Die Einrichtung eines solchen VPN erfordert allerdings eine intensivere Beschäftigung sowohl mit den zugrunde liegenden Konzepten als auch mit den benötigten Konfigurationsdateien; wir wollen dieses Thema deshalb liebend gern der spezielleren Literatur überlassen[5].

Viel einfacher zu handhaben ist das Programm OpenVPN, das virtuelle private Netze für die unterschiedlichsten Anwendungsfälle realisieren kann, von der einfachen Vernetzung zweier Rechner über eine unsichere Verbindung hinweg bis hin zur Anbindung einer großen Zahl von Außendienstler-Laptops an einen zentralen Firmenserver. Dass es nicht nur für Linux, sondern auch für Windows- und andere unixoide Betriebssysteme verfügbar ist, wollen wir hier nur am Rande vermerken.

OpenVPN bietet eine Vielzahl von Betriebsarten und Optionen, die in der zugehörigen Manpage beschrieben sind. Für Standardeinsatzfälle lässt es sich leicht konfigurieren – wie wir nachfolgend beweisen wollen –, und da es nur Verbindungen über einen einzelnen UDP- (oder notfalls auch TCP-) Port aufnimmt (standardmäßig 1194), ist es ohne große Umstände über Gateways und durch Firewalls hindurch benutzbar, selbst mit Address Translation.

Im Gegensatz zu vielen anderen Open-Source-VPNs gilt OpenVPN als sicher und zuverlässig. Gegenüber dem zuvor beschriebenen Secure-Shell-VPN bietet es einige Vorteile, die den etwas höheren Einrichtungsaufwand rechtfertigen können:

- Es benötigt keine Login-Kennung auf der Gegenseite, die gegebenenfalls zusätzlich gegen Missbrauch (oder in manchen Anwendungsfällen auch gegen reguläre Benutzung) zu schützen wäre.
- OpenVPN kann das virtuelle Netz automatisch aufbauen, sobald eine IP-Verbindung zwischen den beteiligten Rechnern zustande kommt.
- Nach Unterbrechungen kann die Verbindung selbsttätig wieder aufgenommen werden.
- OpenVPN ist vielseitiger und flexibler zu konfigurieren.

[5] Ralf Spenneberg, VPN mit Linux. Addison-Wesley, 2003, ISBN 3-8273-2114-X.

Natürlich verschlüsselt OpenVPN sämtliche Übertragungen. Die Partner können einen gemeinsamen, geheim gehaltenen Schlüssel benutzen (Static-Key-Modus) oder mit kryptografischen Verfahren für jede Verbindung einen zufälligen Schlüssel aushandeln, müssen sich dann aber mit digital signierten Zertifikaten gegenseitig ihre Identität beweisen (TLS-Modus; TLS steht für »Transport Layer Security« und war in früheren Versionen unter dem Namen SSL bekannt).

Am sinnvollsten erscheint uns für normale Anwendungen die Benutzung des TLS-Modus in Verbindung mit selbst erstellten Zertifikaten sowie eines geheimen Schlüssels (»TLS authentication key«), der lediglich dazu dient, Annäherungsversuche Unberechtigter zu entlarven und nicht mit jedem dahergelaufenen Hacker in den ziemlich rechenaufwendigen Schlüsselaushandelungsprozess einzutreten[6].

Für diese Einsatzart sind einige Vorbereitungen erforderlich, die beim Lesen der Manpage und der weiteren mitgelieferten Dokumentation umfangreicher und schwieriger erscheinen, als sie tatsächlich sind. Um diese Behauptung zu untermauern, zeigen wir Ihnen die notwendigen Schritte, um eine OpenVPN-Instanz für ein einfaches Szenario einzurichten: Ein Rechner (`hugo`) spielt die Serverrolle; er kann im Büro oder im häuslichen Netzwerk installiert sein und regelt (über eine Netfilter-Firewall) den Zugang zum Internet (z. B. über DSL). Der andere Rechner kann sich über das Internet einloggen, wie wir es bereits im vorigen Abschnitt für das SSH-VPN skizziert haben.

Vorbereitungen

In den gängigen Linux-Distributionen sollte OpenVPN als eigenständiges Paket enthalten sein. Wenn es nicht oder nur in einer veralteten Version verfügbar sein sollte, können Sie unter `http://openvpn.net/` die aktuellen Quellen (und ggf. auch Pakete) zum Download finden und müssen die Software gegebenenfalls selbst übersetzen; dazu wird eine OpenSSL-Installation benötigt. Wir beziehen uns im Folgenden auf die OpenVPN-Version 2.0, die auch eine Reihe von sehr nützlichen Scripts mitbringt, die den Aufruf des Programms `openssl` zur Generierung und Verwaltung digitaler Zertifikate vereinfachen. Eine detailliertere Beschreibung dieses Easy-RSA genannten Werkzeugs finden Sie unter `http://openvpn.net/easyrsa.html`.

Die weiteren Vorbereitungen für den Einsatz von OpenVPN einschließlich der Schlüssel- und Zertifikaterzeugung können ohne weiteres unter einer normalen Benutzerkennung ausgeführt werden. Erst zur Installation von Dateien im Konfigurationsverzeichnis (meist `/etc/openvpn`) werden root-Rechte benötigt. Deswegen beginnen wir im Home-Verzeichnis des Benutzers `victor`, wo wir ein Unterverzeichnis anlegen, das in Anlehnung an den Namen des künftigen OpenVPN-Servers (`hugo`) hier im Beispiel `hugo-ca` genannt wird:

```
victor@hugo:~$ mkdir ~/hugo-ca
victor@hugo:~$ cd ~/hugo-ca
victor@hugo:~/hugo-ca$
```

[6] Diese vorgeschaltete Authentifizierung verhindert auch, dass ungültige Anfragen mit gefälschter Absenderadresse beantwortet werden und dadurch ein unbeteiligter Dritter genervt wird (»amplification attack«).

Sodann sollte die Datei `vars`, die mit der aktuellen Version von Easy-RSA als Muster ausgeliefert wird, in das Arbeitsverzeichnis kopiert werden (der hier benutzte Pfadname gilt für ein Debian-System). Die Datei enthält Variablendefinitionen in Shell-Syntax, die Sie an Ihren Einsatzfall anpassen sollten. Solange Sie später ein isoliertes, eigenes OpenVPN betreiben, sind die Angaben zu `KEY_COUNTRY` bis `KEY_EMAIL` völlig beliebig; die Werte sollten lediglich insoweit eindeutig sein, dass es nicht zu Verwechslungen mit Zertifikaten kommt, die für andere Zwecke erstellt wurden. Wir glauben, mit unseren Angaben dieses Ziel erreicht zu haben – denken Sie sich irgendetwas aus!

Hier ist unsere Fassung der Datei `vars`, die mit Rücksicht auf den Umfang (und Preis) dieses Buches ihrer Kommentarzeilen beraubt wurde:

```
export EASY_RSA="/usr/share/doc/openvpn/examples/easy-rsa/2.0"
export OPENSSL="openssl"
export PKCS11TOOL="pkcs11-tool"
export GREP="grep"
export KEY_CONFIG=‘$EASY_RSA/whichopensslcnf $EASY_RSA‘
export KEY_DIR="/home/victor/hugo-ca/keys"
echo NOTE: If you run ./clean-all, I will be doing rm -rf on $KEY_DIR
export KEY_SIZE=2048
export CA_EXPIRE=3650
export KEY_EXPIRE=3650
export KEY_COUNTRY="DE"
export KEY_PROVINCE="NRW"
export KEY_CITY="Entenhausen"
export KEY_ORG="Netzwerk-Buch"
export KEY_EMAIL="admin@beispiel.org"
```

In der ersten Zeile ist der Name des Verzeichnisses einzusetzen, aus dem Sie die `vars`-Datei geholt haben. Für die Variable `KEY_DIR` sollte unbedingt ein absoluter Pfadname angegeben werden, der auf das Unterverzeichnis `keys` des Arbeitsverzeichnisses zeigt. Mit den Werten `3650` wird der Gültigkeitszeitraum der später erzeugten Zertifikate auf (knapp) 10 Jahre festgelegt; wenn Sie dieses Buch noch länger benutzen wollen, können Sie auch einen höheren Wert eintragen. Uns wäre es aber lieber, Sie würden rechtzeitig eine Neuauflage kaufen, in der wir dann beschreiben, wie Sie die Zertifikate erneuern!

Schlüsselgenerierung

Nachdem Sie zur Sicherheit eine neue Shell gestartet haben, lassen Sie die Shell-Variablen aus der Datei `vars` in dieser Shell setzen, löschen sicherheitshalber alle störenden Dateien (von denen es beim ersten Versuch keine geben sollte), setzen Kaffee (oder Tee[7]) auf und lassen die Parameter für eines der benutzten kryptogra-

[7] Gerne einen Darjeeling, wenn Sie uns einladen möchten. Mit frischer Milch.

fischen Verfahren (das Diffie-Hellman-Verfahren für sicheren Schlüsselaustausch) generieren:

```
victor@hugo:~/hugo-ca$ bash
victor@hugo:~/hugo-ca$ . vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on ./keys
victor@hugo:~/hugo-ca$ $EASY_RSA/clean-all
victor@hugo:~/hugo-ca$ $EASY_RSA/build-dh
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
.........+.................................................
............................................................
.+......+...................................................
```

(Es stimmt: Das dauert wirklich lange, und nur das Erscheinen der Punkte lässt uns hoffen, dass die Primzahlgenerierung mal irgendwann fertig wird … Trinken Sie jetzt in Ruhe den Kaffee oder Tee.)

```
.....................+.............................++*++*
victor@hugo:~/hugo-ca$ echo juchhu, geschafft
juchhu, geschafft
victor@hugo:~/hugo-ca$ $EASY_RSA/pkitool --initca
Using CA Common Name: Netzwerk-Buch CA
Generating a 2048 bit RSA private key
............+++
............................................................
............................................................
.............................................+++
writing new private key to 'ca.key'
-----
victor@hugo:~/hugo-ca$
```

Nun ist die Certificate Authority (CA), die »Behörde«, die für Ihr VPN die Zertifikate ausstellen wird, initialisiert. Sie können das überprüfen, indem Sie in das neu erstellte Unterverzeichnis `keys` schauen: Dort finden Sie die Datei `ca.crt`, die das Wurzelzertifikat enthält, und `ca.key`, den geheimen Schlüssel, der dazu dient, die Echtheit der CA zu beweisen und neue Zertifikate auszustellen, mit denen sich die VPN-Teilnehmer gegenseitig ausweisen können. Dieser Schlüssel wird ohne Leserecht für andere Kennungen erzeugt, er sollte auch keinem Fremden jemals zugänglich werden – sonst könnte sich ein Bösewicht in Ihr VPN einschleichen, Datenverkehr mitlesen und sich unberechtigt als Teilnehmer anmelden. Es ist möglich, den Schlüssel zusätzlich durch ein Passwort zu schützen, worauf wir aber hier verzichten wollen.

Zuerst wird ein Zertifikat für den Server, dann eines für den Client erzeugt:

```
victor@hugo:~/hugo-ca$ $EASY_RSA/pkitool --server hugo
Generating a 2048 bit RSA private key
...................+++
..............+++
writing new private key to 'hugo.key'
-----
Using configuration from
/usr/share/doc/openvpn/examples/easy-rsa/2.0/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'DE'
stateOrProvinceName   :PRINTABLE:'NRW'
localityName          :PRINTABLE:'Entenhausen'
organizationName      :PRINTABLE:'Netzwerk-Buch'
commonName            :PRINTABLE:'hugo'
emailAddress          :IA5STRING:'admin@beispiel.org'
Certificate is to be certified until Oct 13 13:46:49 2016 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
victor@hugo:~/hugo-ca$ $EASY_RSA/pkitool theodor
Generating a 2048 bit RSA private key
..+++
.............................................................
.........................................+++
writing new private key to 'theodor.key'
-----
Using configuration from
/usr/share/doc/openvpn/examples/easy-rsa/2.0/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'DE'
stateOrProvinceName   :PRINTABLE:'NRW'
localityName          :PRINTABLE:'Entenhausen'
organizationName      :PRINTABLE:'Netzwerk-Buch'
commonName            :PRINTABLE:'theodor'
emailAddress          :IA5STRING:'admin@beispiel.org'
Certificate is to be certified until Oct 13 13:50:17 2016 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
victor@hugo:~/hugo-ca$
```

Beachten Sie bitte, dass bei der Generierung des serverseitigen Zertifikats die Option `--server` angegeben werden muss; durch die Unterscheidung zwischen Client- und Serverzertifikaten sollen Angriffe durch einen »man in the middle« unterbunden werden, d. h. durch einen böswilligen Dritten, der sich in Ihre Verbindung einklinkt.

Im Verzeichnis `keys` finden Sie nun die Zertifikate unter den Namen `hugo.crt` (für den Server) und `theodor.crt` (für den Client). Die zugehörigen Schlüsseldateien `hugo.key` und `theodor.key` sind wiederum geheim zu halten!

Unter strenge Geheimhaltung fällt auch der TLS-Auth-Schlüssel, der Unbefugte an der Kontaktaufnahme hindern soll und vom `openvpn`-Programm selbst generiert werden kann:

```
victor@hugo:~/hugo-ca$ openvpn --genkey --secret ta.key
victor@hugo:~/hugo-ca$
```

`--genkey` veranlasst das Programm, einen frischen Schlüssel in die nach `--secret` angegebene Datei zu schreiben und sich dann zu beenden. Sie können stattdessen auch einen frei gewählten String als Schlüssel auswählen und auf Client- und Serverseite hinterlegen, aber der vom Programm erzeugte Zufallsstring ist unvorhersagbar und daher sicherer als alles, was Sie sich (oder wir uns) ausdenken könnten.

Wenn Sie zu Beginn (auf S. 282) unseren Rat befolgt und eine neue Shell gestartet haben, können Sie diese nun mit `exit` beenden; damit sind alle Variablenzuweisungen aus der Datei `vars` aus dem Shell-Gedächtnis verschwunden.

Kopieren Sie nun `ca.crt`, `dh2048.pem`, das Serverzertifikat `hugo.crt`, den Serverschlüssel `hugo.key` und den TLS-Auth-Schlüssel `ta.key` in das Verzeichnis `/etc/openvpn` auf dem künftigen OpenVPN-Server und analog `ca.crt`, `dh2048.pem`, `theodor.crt`, `theodor.key` und `ta.key` in das gleichnamige Verzeichnis auf dem Client. (Bei mindestens einer dieser Aktionen werden Sie Dateien auf einen anderen Rechner kopieren müssen; verwenden Sie hierzu die Secure Shell oder eine Diskette oder einen USB-Stick, damit die Schlüssel vertraulich bleiben! Ein benutztes Speichermedium sollte anschließend mit binären Nullen überschrieben und neu formatiert werden!)

Anschließend liegen die erzeugten (und einige Hilfs-)Dateien weiterhin im Verzeichnis `keys` herum. Entweder sichern Sie diese gut gegen unbefugten Zugriff, oder – falls Sie nicht mit dem Auftauchen weiterer Clients rechnen müssen, für die weitere Zertifikate auszustellen wären – Sie brennen das gesamte Arbeitsverzeichnis mit der `vars`-Datei und dem Verzeichnis `keys` auf eine CD. Löschen Sie dann sämtliche Daten der CA von der Festplatte. Sie brauchen dann nur noch die CD gut verschlossen aufzubewahren, um vor digitalen Panzerknackern sicher zu sein.

Clientkonfiguration

Auf dem Clientrechner, also beispielsweise dem Laptop, legen Sie nun eine Konfigurationsdatei (wiederum im Verzeichnis `/etc/openvpn`) an, deren Namen Sie aus dem Hostnamen des Servers (hier `hugo`) und der Endung `.conf` zusammensetzen. Die

Eintragungen in dieser Datei entsprechen den Kommandozeilenoptionen des Programms `openvpn` und sind daher auch in dessen Manpage beschrieben, nur ist das doppelte Minuszeichen vor jedem Optionsnamen wegzulassen.

Zeilen, die mit # oder einem Semikolon beginnen, gelten als Kommentarzeilen und können, ebenso wie Leerzeilen, zwecks Verschönerung und besserer Wartbarkeit nach Belieben eingefügt werden. Wir haben die Konfigurationsdatei durch Kommentarzeilen in drei Teile gegliedert: Im ersten werden einige grundlegende Einstellungen vorgenommen, die das Verhalten von OpenVPN steuern, dass nämlich das Programm hier die Clientrolle übernehmen und mit welchem entfernten (»remote«) Partner es Kontakt aufnehmen soll.

Auf die Angabe der Spezifikationen `proto udp` und `port 1194` haben wir verzichtet, weil dies sinnvoll gewählte Voreinstellungen sind. Mit `dev tun` wird verlangt, die Kommunikation über virtuelle Netzwerkschnittstellen vom Typ `tun` abzuwickeln; alternativ kann durch `dev tap` eine Verbindung auf Ethernet-Ebene verlangt werden.

```
# Rollen, Protokolle etc.
# -----------------------
client
nobind
remote hugo
dev tun

# Sicherheitseinstellungen: TLS und Preshared Key
# -----------------------------------------------
tls-client
dh /etc/openvpn/dh2048.pem
ca /etc/openvpn/ca.crt
cert /etc/openvpn/paddy.crt
key /etc/openvpn/paddy.key
tls-auth /etc/openvpn/ta.key 1
ns-cert-type server

# allgemeine Einstellungen
# ------------------------
comp-lzo
ping 15
verb 3
```

Der zweite Teil enthält die Namen der zu benutzenden Schlüssel- und Zertifikatdateien. Grundsätzlich sollten die Pfadnamen in den Konfigurationsdateien absolut angegeben werden, damit sie nicht als relative Adressen zum momentanen Arbeitsverzeichnis interpretiert werden, wenn Sie das Programm mit `openvpn server` oder `openvpn <clientname>` direkt von der Kommandozeile aus starten.

Da die TLS-Authentisierung asymmetrisch abläuft, ist auf der Clientseite zur Option `tls-auth` eine Eins, auf der Serverseite eine Null hinzuzufügen.

Die Einstellungen im letzten Abschnitt verlangen eine Kompression der übertragenen Daten mit dem LZO-Algorithmus, regelmäßige Pings in längeren Sendepausen (damit eine dazwischen liegende Firewall nicht die Verbindung kappt) und eine gewisse mittlere Geschwätzigkeit, sodass die wichtigeren Ereignisse im Syslog nachvollzogen werden können.

Serverkonfiguration

Die serverseitige Konfiguration enthält – wen wundert's – anstelle der `client`- eine `server`-Zeile, die zugleich Adresse und Netzmaske des virtuellen Netzes enthält. Im Servermodus wird sich OpenVPN nach dem Lesen der Konfiguration entspannt zurücklehnen und auf Verbindungen warten, die von seinen Artgenossen draußen in der Welt aufgenommen werden. In dem spezifizierten virtuellen Netz wird sich der Server die traditionelle Gateway-Adresse Nummer 1 nehmen, also hier die 192.168.44.1, und den Clients wird er zufällige weitere Adressen zuweisen.

Damit die Clients nicht mit jeder neuen Verbindung eine andere Adresse erhalten, können zum einen die Zuweisungen in der mit `ifconfig-pool-persist` benannten Datei gespeichert werden, sodass derselbe Client beim nächsten Mal möglichst wieder die gleiche Adresse erhält. Zudem kann man in dem als `client-config-dir` vereinbarten Verzeichnis für einzelne Clients separate Konfigurationsdateien hinterlegen, die z. B. die Adresse im virtuellen Netz vorgeben oder Routing-Einstellungen enthalten, die auf dem Client nach erfolgreicher Verbindungsaufnahme vorgenommen werden sollen. Die Dateien in diesem Verzeichnis werden über den Namen identifiziert, der im Zertifikat des Clients als Common Name hinterlegt ist. Lautet dieser beispielsweise `theodor` und steht in der Datei `/etc/openvpn/ccd/theodor`

```
ifconfig-push 192.168.44.12 192.168.44.1
```

(wobei `192.168.44.1` hier die Adresse des Servers im VPN ist), so erhält der Client immer die Adresse 192.168.44.12 zugewiesen. Es ist nicht nur beruhigend, dies im Vorhinein zu wissen, sondern es kann auch ausgesprochen nützlich sein, die Client-Adressen fest zu vergeben: Dadurch ist es möglich, auf dem Server individuelle Firewall-Regeln für den Client festzulegen, die diesem (und keinem anderen) die Nutzung bestimmter Portnummern beim Zugriff auf den Server oder für das Forwarding ins Internet gestatten.

Außerdem ist es so möglich, die Clientadressen in `/etc/hosts`-Dateien aufzunehmen und damit die Clients im virtuellen Netz über gut zu merkende Hostnamen anzusprechen.

Hier ist nun die komplette Serverkonfigurationsdatei `/etc/openvpn/server.conf`:

```
# Rollen, Protokolle etc.
# -----------------------
server 192.168.44.0 255.255.255.0
ifconfig-pool-persist /etc/openvpn/ipp.txt
client-config-dir /etc/openvpn/ccd
```

```
push "redirect-gateway def1"
dev tun

# Sicherheitseinstellungen: TLS und Preshared Key
# -----------------------------------------------
ca /etc/openvpn/ca.crt
cert /etc/openvpn/hugo.crt
key /etc/openvpn/hugo.key
dh /etc/openvpn/dh2048.pem
tls-auth /etc/openvpn/ta.key 0

# allg. Einstellungen
# -------------------
comp-lzo
user nobody
group nogroup
persist-key
persist-tun
keepalive 10 120
status /etc/openvpn/openvpn-status.log
verb 3
mute 20
```

Wir sind Ihnen noch die Erklärung der `push`-Zeile schuldig geblieben: Sofern die Clientkonfiguration dies zulässt, kann der Server dem Client sagen, welche Einstellungen er nach dem Verbindungsaufbau vornehmen soll. Durch ein Push der Anweisung `redirect-gateway` wird der Client eine neue Defaultroute einstellen, die auf das VPN-Gateway zeigt, und damit alle Verbindungen zu IP-Adressen, die nicht direkt oder über explizit vereinbarte Routen erreichbar sind, über das VPN führen. So kann man erreichen, dass ein Rechner nach dem Anschluss an das virtuelle Netz nur noch verschlüsselt kommuniziert, weil sämtliche Datenpakete über das VPN-Gateway geleitet werden. Die Ergänzung `def1` (das letzte Zeichen ist eine Eins!) veranlasst OpenVPN dazu, die Defaultroute nicht zu ersetzen, sondern durch einen Trick umzuleiten. Das ist sehr zu empfehlen, weil Ihr Rechner dann nach einem Absturz der VPN-Verbindung nicht plötzlich ohne Defaultroute dasteht und die Außenwelt nicht mehr erreichen kann.

Das VPN in Betrieb nehmen

Für einen ersten Test können Sie `openvpn` von der Kommandozeile aus starten und als einzige Option, auch wenn die Manpage ein vorangestelltes Argument `--config` verlangt, den Namen der Konfigurationsdatei angeben. Zwar können Sie alle Einstellungen auch über weitere Kommandozeilenoptionen vornehmen, aber es ist sehr ratsam, diese lieber in eine Datei zu schreiben und, falls Sie längere Zeit herumexperimentieren, die einzelnen Versionen dieser Datei und die erzielten Ergebnisse zu dokumentieren, denn nicht immer offenbart sich sofort, dass etwas nicht richtig funktioniert.

In einer Client-Server-Konfiguration, wie wir Sie Ihnen hier vorgestellt haben, beginnt man sinnvollerweise auf der Serverseite. Nach

```
openvpn /etc/openvpn/server.conf
victor@hugo:~$
```

sehen Sie entweder die ersten Fehlermeldungen schon im Terminalfenster, oder es erscheinen welche im Syslog – oder Sie dürfen sich von uns auf die Schulter geklopft fühlen und haben im ersten Anlauf den Server zum Laufen gebracht. Sehen Sie vorsichtshalber in Ihrer lokalen Firewall-Konfiguration nach, ob UDP-Pakete zum Port 1194 zugelassen werden, und fügen Sie gegebenenfalls eine entsprechende Regel hinzu. Das Kommando `netstat` sollte jetzt den aktiven Socket zeigen:

```
hugo:~# netstat -an --inet
Aktive Internetverbindungen (Server und stehende Verbindungen)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
  ...
udp        0      0 0.0.0.0:1194            0.0.0.0:*
  ...
hugo:~#
```

Alsdann kann `openvpn` auch auf der Clientseite gestartet werden. Wenn alles gut geht, wird daraufhin das virtuelle Netz eingerichtet (wenn Sie nicht sofort den Erfolg sehen – *don't panic*, lassen Sie den Programmen einige Sekunden Zeit). Den Erfolg sehen Sie mit Hilfe des Kommandos `ip addr show` oder `ifconfig` (siehe auch Abschnitt A.1.3 auf S. 329), das Ihnen das neue tun- oder tap-Device (mehr dazu im nächsten Abschnitt) mit passender Adresse anzeigen sollte, und anhand von `ip route show`, das eine Route durch den Tunnel auflisten müsste. Die IP-Adresse des tun-Device auf der Serverseite sollte sich nun vom Client aus anpingen lassen, und umgekehrt die clientseitige vom Server aus ebenfalls.

Falls Ihr virtuelles Netz nicht läuft, schauen Sie sich zunächst die von OpenVPN verursachten Syslog-Meldungen auf beiden Seiten an. Vielleicht finden Sie dort einen – nicht unbedingt deutlichen – Hinweis auf eine Fehlersituation, beispielsweise eine fehlende oder nicht lesbare Datei oder einen schlichten Tippfehler in der Konfiguration. Bringt Sie auch das nicht weiter, so ist es ratsam, mit `tcpdump` oder `wireshark` auf beiden Seiten nachzuschauen, ob OpenVPN-Pakete gesendet und auch empfangen werden, ob also das Problem eher auf der Client- oder der Serverseite oder womöglich im Netz dazwischen liegt. (Weitere Tipps zur Fehlersuche finden Sie in Abschnitt 12.3 auf S. 306.)

Haben Sie Ihr VPN erfolgreich eingerichtet, so werden Sie es künftig automatisch starten lassen wollen. Hierzu können Sie die von den Distributionen üblicherweise vorbereitete Init-Datei benutzen (`/etc/init.d/openvpn`); bei Debian-basierten Systemen kann in der Datei `/etc/default/openvpn` mit der Variablen `AUTOSTART` eingestellt werden, welche VPNs gestartet werden sollen.

Wenn das VPN allerdings nur aktiv sein soll, wenn ein bestimmtes Netzwerkinterface konfiguriert wird (z. B. ein PPP- oder WLAN-Device), dann kann der `openvpn`-Aufruf in ein geeignetes Skript verlagert werden, beispielsweise `/etc/ppp/ip-up.local` oder ein Startskript im Verzeichnis `/etc/ppp/ip-up.d`, sofern Ihre Distribution dies vorsieht. Auf Debian-Systemen ist es zudem sehr einfach möglich, den VPN-Start mit der Aktivierung einer Netzwerkschnittstelle zu koppeln, indem ein `openvpn`-Eintrag in die entsprechende »Strophe« der Datei `/etc/network/interfaces` eingefügt wird:

```
allow-hotplug ath0
iface ath0 inet static
        address 192.168.42.12
        netmask 255.255.255.0
        broadcast 192.168.42.255
        gateway 192.168.42.1
        openvpn hugo
```

Dadurch wird, nachdem das Interface `ath0` konfiguriert ist, `openvpn` mit der Konfigurationsdatei `/etc/openvpn/hugo.conf` gestartet. Dann ist freilich der automatische Start beim Hochfahren des Systems zu unterbinden (`AUTOSTART=none` in `/etc/default/openvpn`).

tun- und tap-Modus

OpenVPN kann (wie einige andere VPN-Implementationen auch, beispielsweise die Secure Shell) virtuelle Netzwerkverbindungen im tun- oder im tap-Modus einrichten. Die seltsamen Namen dieser Modi beziehen sich auf den unter Linux üblichen Gerätenamen für das virtuelle Netzwerkinterface.

Der Name »tun« steht für Tunnel. Bei einer Verbindung zwischen zwei tun-Schnittstellen werden nur IP-Pakete (ohne Ethernet-Header) ausgetauscht. Es handelt sich um reine Punkt-zu-Punkt-Verbindungen; sollen Datenpakete über die Gegenstelle hinaus weitergeleitet werden, muss der betreffende Rechner sie routen, also IP-Forwarding eingeschaltet haben. Damit die Pakete überhaupt den Weg durch den Tunnel nehmen, muss zudem auf der Seite des Rechners, der die Daten losschickt, ein Routing-Eintrag existieren, der die Zieladresse dem Tunnel zuordnet.

Die tap-Schnittstellen (engl. *tap*: Anzapfung) hingegen arbeiten auf der Ethernet-Ebene; sie können mehrere Netzwerke wie mit Switches oder Hubs miteinander verbinden, sodass sie ein gemeinsames großes Subnetz bilden. Ein VPN im tap-Modus kann daher einen entfernten Rechner beispielsweise ins Firmennetz einbinden, als wäre er lokal an eine Netzwerkdose angeschlossen. Er kann dementsprechend alle Protokolle (auch die berüchtigten Windows-spezifischen) nutzen wie seine daheimgebliebenen Artgenossen. Dies kann sehr nützlich sein, ist aber etwas schwieriger zu konfigurieren (und zu debuggen), da das serverseitige tap-Device durch eine Software-»Brücke« (eine Bridge) mit dem realen Netz verbunden werden muss. Außerdem wird die VPN-Verbindung stärker (durch Netzwerk-Broadcasts) belastet.

Wir haben uns hier auf die Beschreibung von tun-VPNs beschränkt; wenn Sie tap-Kopplungen einrichten wollen, sollten Sie über gute Kenntnisse der Netzwerkprotokolle verfügen und am besten zunächst mit einem kleinen Testnetz herumexperimentieren, denn es ist ziemlich leicht, durch unglückliche Einstellungen ein ganzes Netz lahmzulegen und sich den Zorn seiner Nutzer zuzuziehen.

11.2.5 vpnc und KVpnc: Clients (nicht nur) für Cisco-VPN

Viele Firmen und insbesondere auch Universitäten setzen VPN-Konzentratoren der Firma Cisco ein, die ein eigenes Protokoll zur Realisierung des virtuellen Netzes benutzen. Dazu gibt es ein proprietäres Clientprogramm, das keine freie Software ist, aber z. B. von Studierenden der Hochschulen, die ein Cisco-VPN einsetzen, üblicherweise kostenfrei heruntergeladen und eingesetzt werden kann. Viele Anwender bevorzugen jedoch das Open-Source-Programm vpnc, das sich als Client ebenfalls mit den Cisco-Servern verbinden kann, zumal es in vielen Linux-Distributionen als fertiges Paket enthalten ist. Es ist ein reines Kommandozeilenprogramm; KDE enthält unter dem Namen KVpnc eine grafische Oberfläche zu vpnc, die zugleich auch einige andere VPN-Clients (unter anderem OpenVPN) fenstergerecht präsentiert.

vpnc

Das Programm vpnc erwartet seine Konfigurationsdaten im einfachsten Fall in der Datei `/etc/vpnc.conf`. Hier ein Beispiel:

```
IPSec gateway vpn.beispiel.org
IPSec ID Galaxy
IPSec secret ArDent42
Xauth username victor
```

Die Angabe des Gateways (als Rechnername oder IP-Adresse) ist erforderlich, damit überhaupt eine Verbindung zum Cisco-Server aufgebaut werden kann. Die ID (hier `Galaxy`) und das sogenannte Gruppen-Passwort (das dem Schlüsselwort `secret` folgt) erfährt man vom Betreiber des Servers; die Kennung und das dazugehörige persönliche Passwort, mit denen man sich anmeldet, werden meist individuell vergeben und stimmen oft mit der vom Rechenzentrum vergebenen Login-Kennung überein. Wir haben hier bewusst darauf verzichtet, das Benutzer-Passwort mit in die Konfigurationsdatei aufzunehmen – nicht, weil wir Ihnen misstrauen würden, sondern weil wir es lieber nicht im Klartext auf der Festplatte hinterlegen wollen. So fragt vpnc beim Start jedes Mal nach dem Passwort, und im Interesse der besseren Sicherheit geben wir es dann manuell ein. Übrigens sollte die Datei `/etc/vpnc.conf` auch nicht mit allgemeinem Leserecht eingerichtet werden!

Der Aufbau der VPN-Verbindung erfolgt mit dem Kommando `vpnc-connect`, das vpnc als Dämon im Hintergrund startet und nach erfolgreichem Aufbau des VPN-Tunnels die Defaultroute so ändert, dass der gesamte Datenverkehr (außer dem zum VPN-Server selbst!) durch den Tunnel geleitet wird. Das Gegenstück zu diesem Kommando heißt `vpnc-disconnect`. Es baut den Tunnel wieder ab und stellt

die ursprüngliche Defaultroute wieder her. Da bei diesen Aktionen die Netzwerkanbindung des Rechners verändert wird, erfordern die vpnc-Kommandos root-Rechte.

KVpnc

Wenn Sie das Programm KVpnc erstmalig starten (es benötigt ebenfalls root-Rechte und fragt deshalb beim Start nach dem root-Passwort), müssen ihm die Zugangsdaten für Ihre VPN-Verbindung(en) mitgeteilt werden. Sie können hierzu den Assistenten für ein NEUES PROFIL aufrufen, der über ein eigenes Icon erreichbar ist, und die Daten in die daraufhin erscheinenden Menüs eingeben: Als VPN-Typ wählen Sie CISCO, und nach DATEN MANUELL EINGEBEN werden letztlich die gleichen Daten verlangt, die auch in der vpnc-Konfigurationsdatei zu hinterlegen waren. Sie können auch hier wählen, ob die Passwörter auf der Festplatte gespeichert werden sollen – der automatische Aufruf der »elektronischen Brieftasche« KWallet sorgt hier für mehr Sicherheit.

Meist stellen die Betreiber eines Cisco-VPN-Konzentrators auch vorgefertigte Parameterdateien in einem herstellerspezifischen (Text-)Format zur Verfügung: sogenannte PCF-Dateien. Steht Ihnen eine solche zur Verfügung, so können Sie die Einrichtung Ihres VPN-Zugangs ein wenig vereinfachen; wählen Sie im KVpnc-Hauptfenster unter PROFIL den Menüpunkt CISCO-PCF IMPORTIEREN aus, und geben Sie im daraufhin erscheinenden Popup-Fenster den Namen der PCF-Datei an. Dies sollte mit der Meldung `Import von importedProfile war erfolgreich` im nächsten Popup quittiert werden. Im nachfolgenden Fenster PROFILE VERWALTEN können Sie nun noch den Namen oder die IP-Adresse des VPN-Gateways angeben (Abb. 11.3).

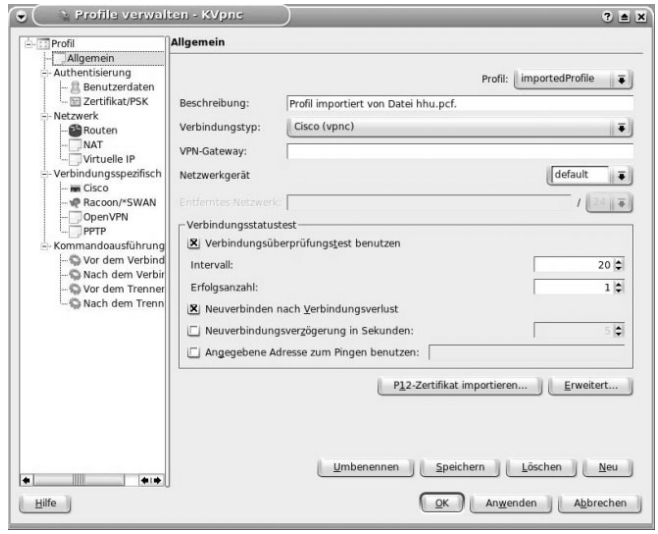


Abbildung 11.3: KVpnc: Profile verwalten

Abbildung 11.4: KVpnc: Einstellungen/Benutzerdaten

Außerdem empfiehlt es sich, das erstellte Profil umzubenennen. Standardmäßig heißt es IMPORTEDPROFILE; unter einem sprechenderen Namen werden Sie es später besser wiedererkennen.

Wählen Sie nun im Einrichtungsfenster unter PROFIL/AUTHENTISIERUNG den Punkt BENUTZERDATEN, und geben Sie Ihre Kennung und Ihr Passwort ein (Abb. 11.4). So können Sie anschließend das Profil im Hauptfenster auswählen und mit dem Button VERBINDEN (oder CONNECT) die VPN-Verbindung starten. KVpnc nistet sich im Systemabschnitt der KDE-Kontrollleiste ein und zeigt dort den Verbindungsstatus an; durch einen Rechtsklick auf das Icon erscheint das Kontextmenü mit einer Option zum Verbinden bzw. zum Trennen der Verbindung.

11.3 Sicherheitstipps

11.3.1 Verräterischer Swapspace

Passwörter und andere vertrauliche Daten können, wenn der Arbeitsspeicher mal knapp wird, im Klartext lesbar im Swapspace erscheinen. Zwar sind sie dort nicht eben leicht zu finden, aber eine geschickte Suche kann sie schnell zutage fördern. Nur wenige Programme wie z. B. GnuPG treffen Vorsorge dagegen und verhindern, dass die benutzten Speicherbereiche in den Swapspace ausgelagert werden.

Das ist besonders bei Laptop-Computern kritisch, die leicht in unbefugte Hände fallen, ob durch Diebstahl oder auch nur vorübergehend in einem unbeobachteten Moment. Wenn der Besitzer dieser Hände Sie ausspionieren will, ist es für ihn ein Leichtes, den Laptop von einer Rettungs-CD zu booten und den Swapspace auf ein

externes Medium (CD, USB-Stick) zu kopieren, um ihn später in aller Ruhe nach sensiblen Daten zu durchforsten.

Deshalb empfiehlt es sich, gerade bei Laptops den Swapspace per (Zufalls-)Passwort verschlüsseln zu lassen oder – bei reichlich vorhandenem Arbeitsspeicher und hohem Sicherheitsbedürfnis – ganz abzuschalten. Die Verschlüsselung lässt sich bei den meisten Distributionen während der Installation oder über das Administrationstool einrichten; bei Debian(-basierten) Systemen genügt es, in die Datei `/etc/crypttab` eine Zeile

```
cswap /dev/hda2 /dev/random swap
```

einzutragen (wobei `/dev/hda2` durch den Namen der tatsächlich für den Swapspace zu verwendenden Partition zu ersetzen ist). Im Eintrag für `swap` in der Datei `/etc/fstab` braucht dann nur noch der Gerätename – im Beispiel `/dev/hda2` – durch den Namen der verschlüsselnden Gerätedatei `/dev/mapper/cswap` ersetzt zu werden. Da der benutzte Schlüssel bei jedem Systemstart neu ausgewürfelt und nur im Arbeitsspeicher aufbewahrt wird, geht er beim Herunterfahren oder Ausschalten des Laptops unwiederbringlich verloren, und niemand kann mit den Daten mehr etwas anfangen. Allerdings funktioniert bislang dann auch das »Suspend to Disk« nicht mehr, da dies beim Hochfahren des Systems den alten Systemzustand aus dem Swapspace zu rekonstruieren versucht.

11.3.2 Virtuelle Gefängnisse

Es gibt in einem normalen Linux-System keinen Schutz gegen Beobachtung durch einen anderen User, der volle root-Rechte besitzt; und auch gegen Manipulationen seitens eines solchen Unholds kann man sich letztlich nicht schützen. Nur in virtuellen Linux-Instanzen oder in einem durch SELinux besonders geschützten System können die Rechte des root-Benutzers so weit beschnitten werden, dass er die Sicherheit des Systems und der Benutzer-Accounts nicht gefährden kann. Wenn Sie also weitestgehende Sicherheit gegen Hacker erzielen wollen, die – womöglich über das Internet zugreifend – »dank« bekannt gewordener Systemfehler root-Rechte erlangt haben, sollten Sie den Einsatz von SELinux oder einer Virtualisierungslösung in Erwägung ziehen.

Speziell hinsichtlich der Netzwerkanbindung virtueller Linux-Maschinen sind einige Besonderheiten zu beachten, die durch die jeweilige technische Realisierung bedingt sind:

- Einfaches `chroot` virtualisiert die Netzwerk-Ressourcen nicht und bringt daher nur wenige Vorteile; es erschwert nur den Zugriff auf das übrige System, wenn ein im chroot-»Gefängnis« laufender Prozess gehackt wurde.
- Linux-VServer-Instanzen[8] sind üblicherweise auf eine einzelne Adresse an einem Interface beschränkt. Damit mehrere Instanzen die gleiche Netzwerkschnitt-

[8] `http://linux-vserver.org/`

stelle nutzen können, müssen ihr entsprechend mehrere Adressen zugewiesen werden. Durch Firewall-Regeln kann für jede dieser Adressen festgelegt werden, welche Ports nutzbar sein sollen; damit lassen sich verschiedene Netzdienste wirksam voneinander trennen.

Allerdings müssen Instanzen, die Netzwerkzugriffe jenseits des normalen IP-Verkehrs ausführen sollen (z. B. DHCP-Server, Benutzung von `tcpdump`), zusätzliche Rechte erhalten und können damit die Trennung der virtuellen Server teilweise unterlaufen.

- Xen[9] ermöglicht es einzelnen virtuellen Maschinen, (PCI-)Netzwerkschnittstellen direkt zu betreiben, wodurch sie sich aus Netzwerksicht genauso verhalten wie ein separater, realer Rechner. Meist werden jedoch virtuelle Schnittstellen benutzt, die in der privilegierten Domäne 0 durch eine Software-Bridge mit dem externen Netz verbunden oder zu diesem hin geroutet werden müssen. Dabei kann durch Firewall-Regeln für eine Entkopplung der Instanzen voneinander gesorgt werden.

11.3.3 Internet-Dämon deaktivieren

In Unix-Systemen ist es seit langem üblich, einen Internet-Dämon `inetd` zu starten, der sich um den Start aller Netzwerkdienste kümmert, die nicht schon beim Systemstart, sondern erst beim Eintreffen einer Verbindungsanforderung initialisiert werden sollen. Beispielsweise gilt dies für den Telnet-Dämon, der ähnlich wie die Secure Shell einem entfernten Nutzer den Shell-Zugang zum System ermöglichen soll, dazu aber ein völlig unsicheres Protokoll verwendet, das nach heutigem Erkenntnisstand nicht mehr guten Gewissens einzusetzen ist.

Anders als der Secure-Shell-Dämon, der sich vom Systemstart an (normalerweise) an den Port 22 bindet und auf Verbindungen wartet, übernimmt für Telnet und ähnliche antike Dienste der `inetd` das Warten. Kommt ein Verbindungswunsch zum Telnet-Port 23 an, so startet `inetd` einen Telnet-Prozess und übergibt ihm die neue Verbindung. Danach wartet er auf neue Anfragen auf Port 23 und allen anderen Ports für sonstige Netzwerkdienste, für die er die gleiche Aufgabe übernehmen soll.

Ursprünglich war dieses Konzept nicht schlecht; es war ressourcensparender, nur einen zentralen Dämon warten zu lassen und weitere nur nach Bedarf zu starten, und in einer zentralen Konfigurationsdatei `/etc/inetd.conf` lassen sich in einigermaßen übersichtlicher Form Voreinstellungen für eine Reihe von Diensten treffen, die alle über den `inetd` verwaltet werden.

Heute sind aber nicht nur Telnet, sondern auch viele andere klassische Unix-Services als unsicher erkannt und durch andere Protokolle ersetzt worden. Dies gilt für alle Dienste, die üblicherweise dem `inetd` zugeordnet werden; sie sollten nur noch in Ausnahmefällen eingerichtet und dann (z. B. durch Firewalling) auf möglichst wenige, vertrauenswürdige Nutzer beschränkt werden. Auf einem normalen

[9] `http://www.cl.cam.ac.uk/research/srg/netos/xen/`

Linux-System wird keine Notwendigkeit hierzu bestehen, und deswegen sollte `inetd` gar nicht gestartet werden.

Am besten deinstallieren Sie das `inetd`-Paket komplett; das Gleiche gilt auch für `xinetd`, einen Nachfolger für `inetd`, der in einigen Distributionen (Red Hat, Fedora) standardmäßig verwendet wird und in anderen ebenfalls verfügbar ist.

11.3.4 IPv6 deaktivieren

Auf die Gefahr hin, dass uns die IPv6-Pioniere steinigen: Sie werden IPv6, den Nachfolger des jetzigen Internet-Protokollstandards IPv4, weder heute, zum Zeitpunkt der Entstehung dieses Buchs, noch in der nahen Zukunft (wenn Sie dieses Buch – ausgelesen und zerfleddert – unter den Wäscheständer legen, damit er nicht wackelt) wirklich nutzen.

Deshalb kann es einstweilen vorteilhaft sein, die IPv6-Unterstützung im Linux-Kernel abzuschalten. Dadurch kann Ihr System – je nach verwendeter Distribution – sogar spürbar schneller werden, weil beispielsweise DNS-Zugriffe über IPv6, die ohnehin zum Scheitern verurteilt sind, von vornherein unterlassen werden. Außerdem kann ein gewisser Sicherheitsgewinn damit verbunden sein, weil unerwünschte IPv6-Verbindungen oft nicht so konsequent gefiltert oder abgeblockt werden, wie man es für IPv4 mittlerweile macht.

Hierzu kommentieren Sie in Ihrer Modulkonfiguration die Zeile

```
alias net-pf-10 ipv6
```

durch Voranstellung eines `#` aus und fügen dafür eine neue Zeile ein:

```
# alias net-pf-10 ipv6
alias net-pf-10 off
```

Je nach Distribution sollten Sie diesen Eintrag in einer der Dateien `/etc/modprobe.d/aliases`, `/etc/modutils`, `/etc/modprobe.conf` oder `/etc/modules.conf` finden. Wenn IPv6 doch demnächst Einzug in Ihr Netz halten sollte, können Sie problemlos den neuen Eintrag entfernen und den alten wieder aktivieren. Wirksam wird die Änderung an dieser Stelle allerdings erst nach dem nächsten Systemstart.

11.3.5 X11-Zugriff schützen

Wenn Sie den Zugriff auf Ihre X-Window-Session über das Netz erlauben (wie beispielsweise in Abschnitt 3.1 auf S. 53 beschrieben), lassen sich einige Arbeiten sehr bequem erledigen: Ein Programm, das auf einem entfernten Rechner startet, kann seine grafische Oberfläche in ein oder mehreren Fenstern auf Ihrem PC öffnen und lässt sich genauso komfortabel bedienen wie ein lokales Programm. Aber leider lässt sich der freie Zugriff auf die Session auch auf vielfältige Weise missbrauchen. Ein besonders perfider Trick besteht darin, von einem anderen Rechner aus ein

großes unsichtbares Fenster zu öffnen, das den ganzen Bildschirm überdeckt und – unbemerkt vom lokalen Benutzer – alle Tastatureingaben und somit auch Passwörter und andere Geheimnisse abfängt.

Wenn Sie ein Programm auf einem anderen Rechner über die Secure Shell starten, können Sie durch die `ssh`-Kommandozeilenoption `-X` oder einen Eintrag `ForwardX11 yes` in der Konfigurationsdatei `/.ssh/config` für Ihre Benutzerkennung veranlassen, dass das Programm über die geschützte, verschlüsselte Secure-Shell-Verbindung Kontakt mit dem lokalen X-Server aufnimmt und Ihnen so seine Fenster präsentieren kann. Damit ist eine Gefährdung durch Fremde weitgehend ausgeschlossen. Allerdings kann ein Bösewicht, der auf dem entfernten Rechner root-Rechte besitzt oder Ihre dortige Kennung gehackt hat, dennoch unberechtigt auf Ihre lokale X-Oberfläche zugreifen. Deshalb sollten Sie selbst den `ssh`-geschützten Zugriff auf den X-Server nur mit Vorsicht benutzen.

Falls widrige Umstände (z. B. schlecht konzipierte Programme) tatsächlich einmal die Freigabe des direkten Zugriffs auf Ihre X-Session verlangen, sollten Sie zumindest durch restriktive Firewall-Regeln für den Zugriff auf den TCP-Port 6000 (und, wenn es mehr als einen X-Server gibt, die nächsten Portnummern, etwa bis 6010) Ihres Rechners verhindern, dass Verbindungen von unbekannten IP-Adressen aus aufgenommen werden.

12 Pflege des Netzwerks

Eigentlich könnten wir uns dieses Kapitel ja sparen: Wenn ein Netzwerk erst einmal läuft, dann läuft es. Meistens jedenfalls. Unserer Erfahrung nach beginnen die Probleme gerade dann, wenn jemand damit beginnt, das Netzwerk zu pflegen, also kleine Verbesserungen, Vereinfachungen, Erweiterungen oder Tuning-Maßnahmen vorzunehmen. Dennoch erscheinen uns einige Hinweise zu Routinearbeiten angebracht, die eher dazu dienen sollen, bestehende Probleme zu erkennen, als neue zu fabrizieren – und wir wollen versuchen, Ihnen für den Fall, dass doch einmal etwas schiefgeht, eine Art von Checkliste zur Fehlersuche an die Hand zu geben.

12.1 Probleme erkennen

Nichts ist schlimmer, als Probleme zu haben, von denen man nichts weiß … Wenn Sie einen (oder mehrere) Linux-Rechner betreiben und für eine sichere Grundkonfiguration sowie regelmäßiges Einspielen von Updates sorgen, sollte es nicht notwendig sein, das System ständig zu beobachten. Zwar wird, falls eine direkte Verbindung zum Internet besteht, das Firewall-Log Unmengen von Port-Scans, Login-Versuchen und anderen unerwünschten Kontaktaufnahmen aus der ganzen Welt registrieren, doch sollte Sie das nicht beunruhigen. Früher war es mal üblich, die Urheber oder deren Provider ausfindig zu machen und sich dort zu beschweren, doch reagiert heute kaum noch jemand auf solche Beschwerden, und die wahren Schuldigen sind ohnehin nicht zu ermitteln.

Kommen allerdings solche Scans von einem Rechner im eigenen (privaten oder Firmen-)Netz, so weist dies in der Regel auf eine Viren- oder Wurminfektion oder die Installation eines Trojanischen Pferdes hin. Entsprechende Hinweise in der Log-Datei sind deshalb ernster zu nehmen.

Auch Hardwarestörungen, übervolle Dateisysteme und Systemfehler kündigen sich oft durch Log-Einträge an. Das passiert allerdings genauso bei einem nicht vernetzten Einzelplatz-PC; deshalb wollen wir auf diese Effekte hier nicht allgemein eingehen, sondern nur, soweit sie das Netz betreffen.

Je mehr Rechner Ihr Netzwerk umfasst, desto umständlicher wird es, deren Syslogs zu überwachen. Sie könnten sich regelmäßig per Secure Shell auf allen Systemen einloggen und die Log-Dateien auf Besonderheiten hin durchsuchen, aber das ist mühsam. Stattdessen können Sie sämtliche Log-Einträge auf einem zentralen Rechner sammeln und dort sichten – oder automatisiert auswerten.

Im Idealfall gibt es einen dedizierten »Log-Host«, der die Syslog-Einträge aller Rechner im eigenen Netzwerk sammelt, selbst aber keine weiteren Services bereitstellt und insbesondere auch nur sehr eingeschränkt erreichbar ist. Dadurch kann gewährleistet werden, dass bei einem Einbruch in ein anderes System der Eindringling nicht seine Spuren verwischen kann; die von ihm verursachten Log-Einträge liegen gut geschützt auf dem Log-Host.

Allerdings kommt eine solche Infrastruktur meist nur für große Netze in Frage. Ein zentrales Logging wird im kleinen Netzwerk nicht so einen großen Sicherheitsgewinn einbringen, aber immerhin die administrativen Arbeiten erleichtern.

Für das zentrale Logging ist Syslog-NG, der Syslog-Dämon der »nächsten Generation«, besser geeignet als sein Vorgänger, der noch standardmäßig von den meisten Distributionen installiert wird. Das Paket `syslog-ng` wird vermutlich auch von der Distribution Ihrer Wahl bereitgestellt; installieren Sie dieses auf dem Rechner, der das zentrale Log führen soll und den wir als Log-Host bezeichnen wollen, auch wenn er diese Aufgabe nicht exklusiv übernehmen wird. Installieren Sie es auch auf den anderen Linux-Systemen, die dem Log-Host ihre Systemmeldungen übermitteln sollen und die wir jetzt Log-Clients nennen wollen.

Entsorgen Sie dafür auf allen beteiligten Rechnern die Pakete `syslogd` und `klogd`. Danach sollte das Logging wie gewohnt (jeweils lokal) weiterlaufen. Editieren Sie nun die Konfigurationsdateien für Syslog-NG, die typischerweise unter `/etc/syslog-ng/syslog-ng.conf` oder `/etc/syslog-ng.conf` installiert sind.

Neben vielen anderen Einstellungen werden in der `syslog-ng.conf`-Datei Quellen für die Meldungen definiert (unter dem Stichwort `source`) und Ziele (jeweils mit einer `destination`-Anweisung) festgelegt, beispielsweise die üblichen Log-Dateien unter `/var/log`. Jeweils eine oder mehrere Quellen werden durch eine `log`-Anweisung mit einem oder mehreren Zielen verbunden, wobei ein (oder auch wieder mehrere) Filter dazwischengeschaltet sind, die entscheiden, welche der von den Quellen gelieferten Meldungen bei den Zielen abgeliefert werden.

Wenn die ausgelieferte Syslog-NG-Konfiguration eines Log-Clients beispielsweise einen Abschnitt

```
log
        source(s_all);
        filter(f_syslog);
        destination(df_syslog);
;
```

enthält, der das Loggen von allen Datenquellen (die zuvor als `s_all` definiert wurden) über den standardmäßigen Filter `f_syslog` in die als Ziel `df_syslog` zugewiesene Datei (meist `/var/log/syslog`) spezifiziert, so können Sie einige Zeilen hinzufügen:

```
destination dn_loghost  tcp("192.168.42.33"); ;
log
```

```
        source(s_all);
        filter(f_syslog);
        destination(dn_loghost);
;
```

Damit wird als neues Logging-Ziel der Log-Host mit der IP-Adresse `192.168.42.33` vereinbart und durch die `log`-Regel dort die gleiche Ausgabe hingeschickt wie zum normalen `df_syslog`-Ziel. Natürlich besagt das Schlüsselwort `tcp`, dass die Übertragung der Daten durch TCP erfolgen soll; das ist zwar aufwendiger, aber zuverlässiger als die ebenfalls mögliche Übermittlung per UDP. Die zu verwendende Portnummer wurde hier nicht angegeben; damit gilt die Voreinstellung 514.

Auch auf der Seite des Log-Hosts muss die Konfiguration ein wenig erweitert werden, damit er das Logging über das Netz überhaupt annimmt. Hierzu fügen Sie der allgemeinen `source`-Spezifikation (die z. B. `s_all` heißt) einen Eintrag für TCP hinzu:

```
source s_all
        internal();
        unix-stream("/dev/log");
        file("/proc/kmsg“ log_prefix("kernel: "));
        tcp();
;
```

Die übrigen Zeilen sollten in dieser oder ähnlicher Form schon vorgegeben sein, lediglich `tcp()` ist als Datenquelle hinzuzufügen.

In einer allgemein vertrauenswürdigen Umgebung können Sie in dem mit `options` eingeleiteten Abschnitt der Konfigurationsdatei die Angabe `keep_hostname(yes);` hinzufügen; dadurch bleibt in den Log-Einträgen das Feld für den Hostnamen unverändert und wird nicht durch die IP-Adresse des Log-Clients ersetzt, was der Lesbarkeit zugute kommt.

Lassen Sie dann mit `/etc/init.d/syslog-ng reload` auf allen beteiligten Rechnern die Syslog-NG-Konfiguration neu einlesen. Danach sollten im Syslog des Log-Hosts auch die Einträge aller Clients erscheinen.

12.2 Schwachstellen suchen

Wir wollen Ihnen nun einige Werkzeuge vorstellen, mit denen man nach Schwachstellen im Netz und in Computern suchen kann. Warum sollte man das tun? In diesen interessanten Zeiten[1] haben offenbar ganze Heerscharen von Crackern und »Script Kiddies« nichts anderes zu tun, als im Internet nach unzureichend geschützten Rechnern zu suchen, die durch eine geeignete Schadsoftware gefügig gemacht werden können und anschließend ferngesteuert werden, um Spam- oder Phishing-Mails zu versenden, illegale Inhalte zu verbreiten oder auf andere Weise den Ruhm

[1] Terry Pratchett: »Interesting Times«. Corgi, 2005

des Angreifers zu mehren. Bevor also Fremde herausfinden, dass Ihr Rechner ein lohnendes Angriffsziel ist, können Sie (im Idealfall) selbst die Schwachstellen herausfinden und beseitigen.

Dazu ist es notwendig, sogenannte Hackertools gegen die eigenen Computer einzusetzen (solange dies noch erlaubt ist). Auf keinen Fall sollten Sie ohne ausdrückliche Absprache fremde Rechner auf Schwachstellen hin untersuchen – damit machen Sie sich bei deren Administratoren garantiert unbeliebt, und möglicherweise ist das sogar strafbar. Und wir müssen auch darauf hinweisen, dass die Anwendung solcher Werkzeuge auf die eigenen Systeme mit Risiken verbunden ist, denn die aggressive Suche nach Verwundbarkeiten kann schon mal einen ganz »normalen« Fehler triggern und zum Absturz des Betriebssystems oder einer Anwendung führen.

12.2.1 Beliebt bei Gut und Böse: nmap

Das Programm nmap gehört zu den sogenannten Mapper-Programmen, die eine »Landkarte« (engl. *map*) eines Netzbereichs erstellen können. Dazu gehören die Entdeckung möglichst aller dort IP-adressmäßig angesiedelten Systeme, deren Identifikation (um welche Hardware, welches Betriebssystem, welche Software handelt es sich?) und, um die Analogie zur Landkarte noch etwas mehr zu strapazieren, das Ausfindigmachen der Sehenswürdigkeiten – aus Hackersicht.

Sehenswert sind aus diesem Betrachtungswinkel alle nachlässig konfigurierten und unzulänglich geschützten Systeme, gleichgültig ob es sich um Computer, Router oder andere Geräte mit IP-Adresse handelt. Aus diesem »touristischen« Interesse heraus sind einige Mapper- und Scan-Programme entstanden, die in der Lage sind, in kurzer Zeit eine große Zahl von Zielsystemen abzugrasen und deren Status zu bestimmen.

Folgende Attribute eines Rechners sind dabei relevant:

- Die Hardwareplattform, weil ein späterer Zugriff auf den Rechner die richtigen Maschinenbefehle nutzen soll.
- Das Betriebssystem, weil gewisse Systeme und Versionen überdurchschnittlich gute Erfolgsaussichten versprechen.
- Die von außen ansprechbaren (»offenen«) Ports, von denen jeder einen theoretisch möglichen Angriffspunkt darstellt.
- Art und Version der Serversoftware, die einen offenen Port bedient, z. B.: Welcher Webserver wird in welcher Version eingesetzt. Wenn für eines der Programme ein Exploit[2] zur Verfügung steht, hat der Hacker sein Ziel schon nahezu erreicht.

Zufälligerweise liefert nmap genau diese Informationen. Das heißt, es versucht, sofern der Anwender es wünscht, Rechner in einem IP-Adressbereich zu finden und eine Vielzahl von Verbindungen zu ihnen herzustellen, um herauszubekommen,

[2] Software zur Ausnutzung einer Schwachstelle im Betriebssystem oder in einem Anwendungsprogramm.

welche Ports offen sind. Aus den Reaktionen des Systems und der mit offenen Ports verknüpften Programme kann nmap anhand von Erfahrungswerten auf das Betriebssystem und die benutzten Serverprogramme schließen. Letztere beantworten oft ohnehin jeden Verbindungswunsch mit einer Selbstauskunft, die den Programmnamen und die Versionsbezeichnung enthält.

Die Dokumentation zu nmap gibt interessante Einblicke in die raffinierten Methoden, mit denen Informationen über das Zielsystem gewonnen werden können, auch wenn dieses wenig Kooperationsbereitschaft zeigt. Für unsere Zwecke kommt es aber nicht darauf an, den Vorgang des Ausspionierens möglichst gut zu verstecken; wir wollen nur wissen, was Fremde über unser System herausfinden können und ob unser Rechner dazu einlädt, ihn zu hacken.

Probieren wir nmap einmal an einem unserer eigenen Systeme aus. Die Kommandozeilenparameter sehen etwas kryptisch aus, deshalb eine kurze Erklärung: `-sT` wählt unter den zahlreichen Methoden, den Rechner abzugrasen, den TCP-Connect-Scan aus, `-A` aktiviert die nach Meinung des Autors fortschrittlichen Funktionen (»Advanced«), die Betriebssystem- und Versionserkennung umfassen. `-T4` stellt eine ziemlich aggressive, also zügige Vorgehensweise ein (wer unberechtigt fremde Rechner scannt, wird langsam arbeiten wollen, um nicht aufzufallen). Die Option `-oA` mit einem nachfolgenden Dateinamenspräfix verlangt die Erstellung von Protokolldateien in mehreren verschiedenen Formaten; sofern sie schon bei einem früheren Lauf erzeugt wurden, sollen sie nicht überschrieben, sondern fortgeschrieben werden (`--append-output`). Am Ende des Kommandos steht schließlich der Name oder die IP-Adresse des Opfers:

```
hugo:~# mkdir -p nmap
hugo:~# nmap -sT -A -T4 -oA nmap/192.168.42.6 --append-output 192.168.42.6

Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2006-11-06 18:05 CET
Warning:  OS detection will be MUCH less reliable because we did not find at
least 1 open and 1 closed TCP port
Interesting ports on 192.168.42.6 (192.168.42.6):
Not shown: 1678 filtered ports
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 4.3p2 Debian 5 (protocol 2.0)
631/tcp open  ipp     CUPS 1.2
MAC Address: 00:0D:88:4A:55:6B (D-Link)
Too many fingerprints match this host to give specific OS details
Service Info: OS: Linux

Nmap finished: 1 IP address (1 host up) scanned in 42.534 seconds
hugo:~#
```

Dieser Scan war nicht besonders ergiebig – die Betriebssystemerkennung hat ein nur wenig spezifisches Ergebnis erbracht (`Linux`), und es wurden nur zwei offene Ports entdeckt (die in Wirklichkeit sogar nur zum lokalen Netz hin geöffnet sind,

von dem aus gescannt wurde). Was auf diesen läuft, ist allerdings sehr deutlich zu erkennen, und damit kann auch auf die eingesetzte Linux-Distribution und deren Versionsstand geschlossen werden. Insgesamt zeigt sich ein Ergebnis, das nicht zum Hacken einlädt (es sei denn, der Hacker hat zufällig ein Exploit für diese Versionen der Secure Shell oder des CUPS zur Hand).

Nun noch ein anderer Versuch, der sich gegen ein etwas betagtes Sun-System richtet:

```
hugo:~# nmap -sT -A -T4 -oA nmap/192.168.42.6 192.168.42.97

Starting Nmap 4.11 ( http://www.insecure.org/nmap/ ) at 2006-11-06 18:10 CET
Interesting ports on 192.168.1.97 (192.168.1.97):
Not shown: 1653 closed ports
PORT      STATE SERVICE          VERSION
22/tcp    open  ssh              SunSSH 1.1 (protocol 1.99)
37/tcp    open  time              (32 bits)
111/tcp   open  rpcbind           2-4 (rpc #100000)
515/tcp   open  printer          Solaris lpd
665/tcp   open  unknown
898/tcp   open  http             Sun Solaris Management Console (Runs
                                  Tomcat webserver)
2049/tcp  open  nfs               2-3 (rpc #100003)
2401/tcp  open  cvspserver       cvs pserver
4045/tcp  open  nlockmgr          1-4 (rpc #100021)
5432/tcp  open  postgresql       PostgreSQL DB
6112/tcp  open  dtspc?
7937/tcp  open  nsrexec           1 (rpc #390113)
7938/tcp  open  rpcbind           2 (rpc #100000)
8000/tcp  open  http-alt?
13722/tcp open  netbackup        Veritas Netbackup java listener
13782/tcp open  VeritasNetbackup?
13783/tcp open  tcpwrapped
27000/tcp open  flexlm           FlexLM license manager
27001/tcp open  flexlm           FlexLM license manager
27002/tcp open  flexlm           FlexLM license manager
27007/tcp open  flexlm           FlexLM license manager
27009/tcp open  flexlm           FlexLM license manager
32774/tcp open  metad             1-2 (rpc #100229)
32775/tcp open  metamhd           1 (rpc #100230)
32776/tcp open  rpc.metamedd      1 (rpc #100242)
32777/tcp open  mdcommd           1 (rpc #100422)
32778/tcp open  status            1 (rpc #100024)
Device type: general purpose
Running: Sun Solaris 9|10
OS details: Sun Solaris 9 or 10
```

```
Uptime 48.346 days (since Tue Sep 19 07:54:45 2006)
Service Info: OS: Solaris

Nmap finished: 1 IP address (1 host up) scanned in 201.115 seconds

hugo:~#
```

Die große Zahl offener Ports erhöht schon rein statistisch das Risiko dafür, dass unter einem von ihnen eine verwundbare Software ansprechbar ist. Einige der Services gelten als sehr komplex (dazu gehören z. B. die RPC-Dienste und Management-Programme), und andere (wie time auf Port 37) dürften ziemlich überflüssig sein. Insgesamt stellt sich dieser Rechner dem Hacker sicher als viel attraktiver dar als der zuvor gescannte.

Es ist also durchaus sinnvoll, die eigenen (!) Computer hin und wieder abzuscannen, um mögliche Schwachstellen zu erkennen. Auf diese Weise können Programme entdeckt werden, die unbeabsichtigt Dienste nach außen anbieten oder die inzwischen gar nicht mehr benutzt werden.

amap

Weniger gut zum »Mappen« von Netzen geeignet, dafür noch etwas ergiebiger bei den eigentlichen Portscans ist das Programm amap. Es kann die gefundenen Services besser identifizieren als nmap, auch wenn die dahinterstehenden Programme nicht selbst ihren Namen und ihre Versionsnummer offenbaren. Wollen Sie einen Ihrer(!) Rechner mit amap untersuchen, beispielsweise den Server mit dem Namen `hugo`, können Sie das Programm so aufrufen:

```
amap -qb1 hugo 1-65535
hugo:~#
```

Mit `1-65535` werden die zu analysierenden Ports vorgegeben – hier wird der ganze mögliche Bereich abgegrast. Die Manpage zu amap listet noch eine Reihe zusätzlicher Optionen auf; die IP-Adressen und Portnummern der Zielrechner können auch direkt einer von `nmap` erstellten Protokolldatei entnommen werden, sodass die beiden Programme sich gegenseitig ergänzen – zum Nutzen des Hackers …

12.2.2 Härtetest für Webserver: nikto

Der Scanner nikto ist speziell zur Aufdeckung von Schwachstellen in der Konfiguration von Webservern und in Webseiten konzipiert. Wenn er auf einen Webserver losgelassen wird, bombardiert er diesen mit HTTP-Anfragen und analysiert sowohl die Antworten des Servers als auch die Inhalte der zurückgelieferten HTML-Seiten. Wenn das Programm mögliche Probleme entdeckt, gibt es einen Hinweis aus:

```
victor@hugo:~$ nikto -h localhost
---------------------------------------------------------------------------
- Nikto 1.35/1.35     -     www.cirt.net
```

```
+ Target IP:       127.0.0.1
+ Target Hostname: localhost
+ Target Port:     80
+ Start Time:      Mon Nov  6 18:41:41 2006
---------------------------------------------------------------------------
- Scan is dependent on "Server“ string which can be faked, use -g to
    override
+ Server: Apache/2.0.55 (Debian) DAV/2 SVN/1.3.2 mod_python/3.1.3
    Python/2.3.5 PHP/4.4.2-1.1 mod_perl/2.0.2 Perl/v5.8.8
+ Allowed HTTP Methods: GET,HEAD,POST,OPTIONS,TRACE
+ HTTP method 'TRACE' is typically only used for debugging. It should be
    disabled. OSVDB-877.
+ PHP/4.4.2-1.1 appears to be outdated (current is at least 5.0.3)
+ mod_perl/2.0.2 appears to be outdated (current is at least 5.8)
+ 1.1  - May be vulnerable to JSP source code exposure. CAN-2002-1148.
+ / - TRACE option appears to allow XSS or credential theft. See
    http://www.cgisecurity.com/whitehat-mirror/WhitePaper_screen.pdf for details
+ /doc/ - The /doc directory is browsable. This may be /usr/doc. (GET)
+ /manual/ - Web server manual? tsk tsk. (GET)
+ /analog/ - This might be interesting... (GET)
+ 2658 items checked - 6 item(s) found on remote host(s)
+ End Time:        Mon Nov  6 18:41:47 2006 (6 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
victor@hugo:~$
```

Sie erkennen an diesem Beispiel, dass der Scan nicht von einem anderen Rechner aus durchgeführt werden muss, sondern auch den `localhost` ansprechen kann, und es werden keine root-Rechte benötigt.

Wir haben die Ausgabe zur Platzersparnis ein wenig gekürzt; der Auszug des Protokolls zeigt aber schon, dass nikto sich bemüht, verständliche Warnhinweise (freilich auf Englisch) auszugeben und teilweise sogar auf Verbesserungsmöglichkeiten oder nützliche Dokumentation hinweist.

Nicht alle der aufgedeckten »Schwachstellen« stellen ernsthafte Gefährdungen dar; nikto übertreibt gern ein wenig. Es ist aber durchaus ratsam, das Scanergebnis einmal aufmerksam durchzusehen und jedem Hinweis nachzugehen. Wenn Sie sich entschieden haben, welche Kritikpunkte Sie abstellen und welche Sie als tolerierbar hinnehmen wollen, können Sie den nikto-Scan in gewissen Abständen wiederholen, um auf neu entstandene Probleme aufmerksam zu werden.

12.3 Fehlersuche bei Netzwerkproblemen

Ganz grob haben wir unsere Checkliste nach aufsteigenden Ebenen des ISO-Schichtenmodells geordnet. Aber dieses Modell lässt sich ohnehin auf heutige IP-Netzwer-

ke nicht wirklich systematisch anwenden[3], und Sie brauchen es für das Folgende nicht zu beachten. Es hat sich aber als sinnvoll erwiesen, bei der Fehlersuche bei den elementaren Netzwerkfunktionen zu beginnen und erst, wenn diese überprüft sind, die darauf aufbauenden höheren Ebenen unter die Lupe zu nehmen.

12.3.1 Funktioniert der Draht?

Dieser Abschnitt ist für uns der schwierigste, denn wir möchten Sie bitten, mit einigen ganz simplen Checks zu beginnen. Damit wollen wir keineswegs Ihre Intelligenz in Zweifel ziehen – natürlich wissen Sie, dass elektrische Geräte nicht funktionieren, wenn der Stecker nicht in der Steckdose steckt –, doch haben wir die Erfahrung gemacht, dass gerade die versierten Administratoren die elementaren Fehlerquellen leicht übersehen, weil ihnen zu jedem Problem gleich eine Menge möglicher und kompliziertester Ursachen einfällt. Think simple!

- Bringen wir es also hinter uns: Sind alle notwendigen Geräte eingestöpselt? Leuchten die grünen Lämpchen alle? (Netzteile gehen meist als Erstes kaputt.)
- Hat die Kommunikation eine Chance? Sind die Kabel alle eingestöpselt, nicht geknickt oder geklemmt?
- Ist die WLAN-Karte richtig im Slot, ist die Antenne dran, läuft der Access-Point?
- Bei DSL/Modem-Problemen: Geht das normale analoge oder ISDN-Telefon noch?

Netzwerkkarten haben oft eine Leuchtdiode, die den Linkstatus anzeigt. Sie leuchtet (manchmal auch bei ausgeschaltetem Rechner), wenn über das Kabel ein Kontakt zu einem funktionierenden Netzwerkinterface am anderen Ende besteht. Leuchtet sie nicht, sollten Sie nach der Ursache fahnden. Meist blinkt die Anzeige auch, wenn Daten empfangen oder gesendet werden; ein Ping sollte als rhythmisches Blinken zu erkennen sein, reger Netzverkehr als unregelmäßiges Flackern.

12.3.2 Sieht Linux den Draht?

Ein Netzwerkinterface muss nicht nur physikalisch funktionieren, Linux muss es auch ansprechen können. Als ersten Check können Sie `ip link show` aufrufen und nachsehen, ob alle vorhandenen Interfaces aufgelistet werden und den erwarteten Namen tragen. Am besten notieren Sie für jeden Rechner bei der Installation die Zuordnung aller MAC-Adressen zu externen Schnittstellen, damit Sie bei später auftretenden Problemen leichter herausfinden, welche Schnittstelle (z. B. nach einem Software-Upgrade) verschwunden ist oder ob die Namen vertauscht wurden.

Vermissen Sie systemseitig ein Interface, das offensichtlich hardwaremäßig vorhanden ist, so ist entweder das zuständige Kernelmodul nicht geladen worden oder es hat in der Initialisierungsphase die Netzwerkkarte nicht erkannt. Welches Modul zuständig ist, lässt sich notfalls durch eine Internetrecherche herausfinden; wenn

[3] Wir glauben, eine einfache Regel erkannt zu haben: Alle Netzanwendungen, die besonders nützlich sind, verletzen das ISO-Modell.

Sie den Namen kennen, können Sie mit dem `modprobe`-Kommando versuchen, es zu laden, und `modinfo <modulname>` zeigt an, welche Parameter dem Modul übergeben werden können.

Wenn das Interface dem System bekannt ist, lassen sich Informationen über seinen Zustand abfragen. Die meisten (aber beileibe nicht alle!) Treibermodule unterstützen die Abfrage durch wenigstens eines der Kommandos `ethtool` (oft in einem separaten, gleichnamigen Paket verfügbar) oder `mii-tool` und liefern eine Reihe nützlicher Daten aus:

```
hugo:~# ethtool eth0
Settings for eth0:
        Supported ports: [ TP MII ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Advertised auto-negotiation: Yes
        Speed: 100Mb/s
        Duplex: Full
        Port: MII
        PHYAD: 32
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: pumbg
        Wake-on: d
        Current message level: 0x00000007 (7)
        Link detected: yes
hugo:~# mii-tool -v eth0
eth0: negotiated 100baseTx-FD, link ok
  product info: vendor 00:00:00, model 0 rev 0
  basic mode:   autonegotiation enabled
  basic status: autonegotiation complete, link ok
  capabilities: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
  advertising:  100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD
  link partner: 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD flow-control
hugo:~#
```

Ist der Link-Zustand als `yes` oder `ok` angegeben, dürfen Sie das als gutes Zeichen werten: Der Treiber hält die Karte für betriebsbereit. Die mit 10, 100 oder 1000 beginnenden Schlüsselwörter beziehen sich auf die möglichen Leitungsgeschwindigkeiten, die das Interface unterstützt (10 Mbit/s, 100 Mbit/s oder 1 Gbit/s). Miteinander verbundene Interfaces versuchen, die höchstmögliche Geschwindigkeit und den besten Übertragungsmodus (FD, *full duplex*, ist besser als HD, *half duplex*) auszuhandeln, und das dürfte auch in Ihrem Interesse liegen.

Treten allerdings gelegentliche Fehler in einer Verbindung auf, die nicht genau zu reproduzieren sind, kann das an einer mangelhaften Leitungsqualität liegen. Vielleicht wird eine Gigabit-Verbindung über defekte oder zu alte Kabel geführt. Bei einer Abfrage der Statistikdaten für das Interface kann sich dieses Problem in einer deutlich von null verschiedenen Fehlerzahl äußern:

```
hugo:~# ip -s link show ath0
3: eth0:  mtu 1500 qdisc pfifo_fast qlen 200
    link/ether 00:05:4f:4c:2b:30 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5263921    26742    3813    0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5807142    48191    4       0       0       0
hugo:~#
```

Bei Kabelverbindungen sollten die Zahlen unter `errors` null oder jedenfalls sehr klein sein; im WLAN sind allerdings höhere Fehlerraten normal.

Sie können mit dem Kommando

```
mii-tool -F 100baseTx-FD
```

oder

```
ethtool -s eth0 autoneg off duplex full speed 100
```

versuchen, die Geschwindigkeit der Leitung auf 100 Mbit/s herunterzudrehen. Wenn die Probleme damit verschwinden, haben Sie die Wahl, ein neues, besseres Kabel zu ziehen oder sich mit der geringeren Datenrate zufriedenzugeben.

Tauchen in der Interface-Statistik Kollisionen (`collsns`) auf, wird die Leitung offenbar im Halbduplexmodus betrieben. Das bedeutet, dass nur immer eine der Stationen Daten senden darf; die andere muss solange den Mund halten und darf erst dann selbst senden, wenn die erste zu Ende gesprochen hat. (Wenn dieser Betriebsmodus Sie an Funkverkehr mit Walky-Talkys erinnert – »Max für Moritz, bitte kommen« –, dann haben Sie das Prinzip verstanden.)

Mit heutiger Hardware ist Vollduplexbetrieb üblich; mit den oben genannten Kommandos kann durch die Angabe von `...-FD` bzw. `duplex full` diese Betriebsweise erzwungen werden. Sie sollte dann auch am anderen Ende fest eingestellt werden.

12.3.3 WLAN: Sieht Linux den Access-Point?

Die Kommandos `ethtool` und `mii-tool` lassen sich nicht auf die (uns bekannten) WLAN-Interfaces anwenden; viele der darüber zu beeinflussenden Parameter haben für das WLAN auch keine Bedeutung. Stattdessen dient das bereits in Ab-

schnitt 2.3.4 auf S. 41 beschriebene Kommando `iwconfig` zur Konfiguration und zur Abfrage des Interface-Status:

```
hugo:~# iwconfig ath0
ath0      IEEE 802.11g  ESSID:"LAGU"
          Mode:Managed  Frequency:2.462 GHz  Access Point: 00:0F:66:E9:99:11
          Bit Rate:54 Mb/s   Tx-Power:18 dBm   Sensitivity=0/3
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality=34/94  Signal level=-61 dBm  Noise level=-95 dBm
          Rx invalid nwid:10095  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:1  Invalid misc:1   Missed beacon:0

hugo:~#
```

Wenn ein Access-Point erkannt und eine Verbindung zu ihm hergestellt wurde, erscheint in der Ausgabe dessen »cell identity«, gewissermaßen seine MAC-Adresse im Funknetz, die aber keineswegs mit seiner MAC-Adresse übereinstimmen muss, die wir im Netzwerk sehen.

Durch die Festlegung der ESSID (Extended Service Set IDentifier) beim Einrichten des WLAN-Interface sollte sich dieses beim richtigen Access-Point (und nicht bei dem Ihres Nachbarn) anmelden. Sollte es dennoch nötig sein, kann mit `iwconfig` die Identität des Access-Points oder die Nummer des gewünschten Sendekanals festgelegt werden. Die Optionen für diese (und eine ganze Reihe weiterer) Einstellungen sind in der Manpage des Kommandos beschrieben.

Die `iwconfig`-Ausgabe enthält einige Informationen zur Qualität der Funkübertragung: Anstelle einer Leitungsgeschwindigkeit findet sich hier die Bitrate; wenn diese zu niedrig ausfällt, sind die Empfangsbedingungen schlecht, und Sie sollten nach Möglichkeit einen günstigeren Standort auswählen. Auch die Link-Qualität und der Signal-Level können helfen, die besten Bedingungen für die Datenübertragung herauszufinden.

12.3.4 Stimmen die IP-Adressen?

Überprüfen Sie sicherheitshalber mit dem Kommando `ip addr show`, ob den Netzwerkschnittstellen korrekte IP-Adressen zugewiesen worden sind. Ein häufiger Fehler liegt darin, dass vergessen wurde, die Netzmaske anzugeben:

```
hugo:~# ip addr show eth0
3: eth0:  mtu 1500 qdisc noqueue
    link/ether 62:b2:ba:45:d4:13 brd ff:ff:ff:ff:ff:ff
    inet 192.168.42.7/32 scope global eth0
    inet6 fe80::60b2:baff:fe45:d413/64 scope link
       valid_lft forever preferred_lft forever
hugo:~#
```

Verräterisch ist die an die IP-Adresse angehängte `32`: Sie zeigt an, dass Linux nur eine einzelne Adresse und kein Subnetz zugewiesen hat. Deshalb ist das eigentlich zu der Adresse gehörende Subnetz 192.168.42.0/24 unerreichbar.

Auch fehlt in der `inet`-Zeile dieser Kommandoausgabe der Eintrag `brd 192.168.42.255`, d. h., es wurde bei der Konfiguration der Schnittstelle keine Broadcastadresse zugewiesen. Einige Anwendungen, die Broadcasts benutzen, werden so nicht funktionieren.

12.3.5 Funktioniert IP?

Nachdem Sie sich von der Existenz und der elektrischen Nutzbarkeit des Kabels (oder der Funkverbindung) überzeugt haben, ist es an der Zeit, die Zustellbarkeit von Datenpaketen zu überprüfen. Auch dabei beginnen Sie am besten mit der einfachsten Variante, den vom `ping`-Kommando verwendeten ICMP-Paketen. Wir haben schon mehrfach durchblicken lassen, was wir davon halten, das Ping durch Firewalls abzublocken – zumindest im lokalen Netz sollte sich jeder von jedem pingen lassen. Das ermöglicht es Ihnen, sowohl die Erreichbarkeit eines Rechners festzustellen als auch die Qualität der Verbindung ganz grob zu beurteilen:

```
victor@hugo:~$ ping 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=1 ttl=64 time=0.795 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=2 ttl=64 time=0.749 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=3 ttl=64 time=0.809 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=4 ttl=64 time=0.754 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=5 ttl=64 time=0.747 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=6 ttl=64 time=0.754 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=7 ttl=64 time=0.758 ms
64 bytes from 192.168.1.11 (192.168.1.11): icmp_seq=8 ttl=64 time=0.757 ms
 STRG + C
--- 192.168.1.11 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 6999ms
rtt min/avg/max/mdev = 0.747/0.765/0.809/0.032 ms
victor@hugo:~$
```

Wenn die in der letzten Spalte angezeigten Laufzeiten zu stark schwanken, deutet das auf Probleme mit der Leitungsqualität oder Überlastung eines dazwischen liegenden Routers, eines Switches oder eines der beteiligten Rechner hin. Führen Sie in Ihrem Netz hin und wieder Pings aus, auch wenn alles bestens läuft, damit Sie ein Gefühl dafür bekommen, welches Verhalten als normal zu gelten hat.

Wenn Aussetzer in der Folge der empfangenen Ping-Echos vorkommen, sollten Sie beginnen, sich Sorgen zu machen. Wir wollen Sie ja nicht beunruhigen, aber bei einigen Netzwerkprotokollen führen Paketverluste direkt zu Datenverlusten, und andere – beispielsweise TCP – versuchen zwar, durch automatische Wiederholung von Paketen Datenverlust zu vermeiden, werden dadurch aber stark verlangsamt.

Bevor Sie sich allerdings durch uns in Aufregung versetzen lassen, weil `ping` Pakete verliert, prüfen Sie erst noch einmal, ob nicht doch eine Firewall schuld ist: Vielleicht sind die ICMP-Pakete zwar zugelassen, aber durch eine `limit`-Regel beschränkt. Werden beispielsweise die ersten fünf Pings akzeptiert und tritt dann eine längere Unterbrechung auf, so liegt dieser Verdacht nahe. Sie könnten in diesem Fall die Firewall-Regeln abändern und den ICMP-Typ `echo-request` von vertrauenswürdigen Rechnern unlimitiert akzeptieren.

Nachbarn

Um festzustellen, ob überhaupt die Kommunikation im lokalen Subnetz funktioniert, können Sie mit `ip neigh show` den ARP-Cache anzeigen lassen. Wenn Sie zuvor alle relevanten Adressen im Subnetz – zumindest durch `ping` – angesprochen haben, sollten Sie die zugehörigen MAC-Adressen im ARP-Cache wiederfinden (selbst, wenn das Ping abgeblockt wurde!):

```
hugo:~# ip neigh show
192.168.42.12 dev eth0 lladdr 00:05:4f:4c:2b:30 REACHABLE
192.168.42.1 dev eth0 lladdr 00:0f:66:e9:99:0f STALE
hugo:~#
```

Näheres hierzu finden Sie im Anhang ab S. 333. An dieser Stelle möchten wir nur darauf hinweisen, dass in der Ausgabe erkennbar ist, an welcher Schnittstelle der jeweilige Nachbarrechner gesehen wurde. In kompliziert strukturierten Netzen sollte überprüft werden, ob diese Informationen den Erwartungen entsprechen.

Wenn die Ausgabe sehr umfangreich ist (was in einem kleinen Netzwerk eigentlich nicht passieren sollte), kann durch die große Anzahl von Einträgen die ARP-Tabelle des Kernels überlaufen. Das heißt, es werden hin und wieder Einträge aus dieser Tabelle gelöscht, die eigentlich noch gebraucht würden, und dadurch können Verbindungen gestört werden. Falls Ihr Rechner unter diesem Problem leidet, können Sie durch Vergrößern der Tabelle einfache Abhilfe schaffen:

```
echo 8192 >/proc/sys/net/ipv4/neigh/default/gc_thresh1
echo 8192 >/proc/sys/net/ipv4/neigh/default/gc_thresh2
echo 8192 >/proc/sys/net/ipv4/neigh/default/gc_thresh3
```

`8192` ist hier ein Beispiel für hinreichend groß gewählte Zahlen.

Routen

Mit dem Kommando `ip route show` können Sie das Routing überprüfen. Das häufigste Problem mit Routen besteht darin, dass die Defaultroute nicht gesetzt wurde (oder auf mysteriöse Weise abhanden kam).

```
hugo:~# ip route show
192.168.42.0/24 dev eth0  scope link
```

```
default via 192.168.42.1 dev eth0
hugo:~#
```

Die Defaultroute (durch das Schlüsselwort `default` gekennzeichnet) gibt an, über welches Gateway alle entfernten Rechner zu erreichen sind. Fehlt sie oder weist sie in die Irre, können nur noch Stationen im lokalen Subnetz erreicht werden.

12.3.6 Ist die Firewall im Weg?

Wenn nicht von vornherein auszuschließen ist, dass aufgetretene Netzwerkprobleme irgendetwas mit Firewall-Einstellungen zu tun haben, sollten Sie das Firewalling genauer unter die Lupe nehmen. Verlassen Sie sich bei der Fehlersuche nicht auf grafische oder Prozedur-Oberflächen, die Ihnen vielleicht die Regelerstellung erleichtert haben; wenn solch ein Tool einen Fehler gemacht hat, wird es bei der Suche nach dem Fehler nicht allzu hilfreich sein. Verwenden Sie lieber das Kommando `iptables` direkt:

```
hugo:~# iptables -L -nv
Chain INPUT (policy ACCEPT 0 packets, 0 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 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source       destination
hugo:~#
```

Wenn Sie eine solche spärliche Ausgabe sehen, sind keine Filterregeln für die Firewall definiert, und somit können hier auch keine Probleme entstehen. Prüfen Sie dann sicherheitshalber noch, ob Address Translation zum Einsatz kommt:

```
hugo:~# iptables -t nat -L -nv
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source       destination

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source       destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source       destination
hugo:~#
```

Ist auch hier nichts vorzufinden, müssen Sie leider Ihre Fehlersuche mit dem nächsten Punkt fortsetzen. Ansonsten können Sie sich der Einfachheit halber dem gängigen Vorurteil anschließen, das Firewalling sei an allen Netzwerkproblemen

schuld … Aber selbst dann ist es ratsam, zunächst die aufgelisteten Firewall-Regeln kritisch auf logische Fehler zu prüfen.

Schauen Sie auch vorsichtshalber nach, ob für einen Rechner, der Routing-Aufgaben ausübt, das Forwarding eingeschaltet ist:

```
hugo:~# cat /proc/sys/net/ipv4/ip_forward
1
hugo:~#
```

Wenn die Überprüfung der Firewall-Regeln zu keinem Erfolg führt oder das (von einem Tool erzeugte) Regelwerk zu kryptisch ist, könnten Sie versuchen, das Firewalling testweise ganz abzuschalten, um zu sehen, ob die Probleme dadurch verschwinden.

Das Firewalling abschalten

Freilich sollten Sie das nur tun, wenn Sie sicher sein können, dass Sie Ihr System dadurch keiner besonderen Gefahr aussetzen. Es sollten keine Dienste laufen, die auf Attacken von außen empfindlich reagieren; vielleicht können Sie während der Tests die Verbindung des Rechners zum Internet kappen. Grundsätzlich sollte ein gut konfiguriertes Linux zwar auch ohne jede Firewall-Regel im Internet überlebensfähig sein, aber Anwender (und auch Administratoren) neigen zum Leichtsinn, wenn sie wissen, dass eine Firewall vor den schlimmsten Angriffen aus dem Netz schützt. Diese Einstellung rächt sich schnell, wenn die Firewall mal nicht wirksam ist.

Damit Sie das Firewalling nicht nur ab-, sondern auch ohne große Mühe wieder einschalten können (wenn die bestehenden Probleme nicht mit der Firewall zusammenhängen), ist es ratsam, die Firewall-Regeln nicht einfach zu löschen. Wenn die Firewall so eingerichtet ist, dass für eine oder mehrere Regelketten die Policy `DROP` gesetzt ist und der erlaubte Netzverkehr durch `ACCEPT`-Regeln herausgepickt wird, können Sie einfach die Policy vorübergehend auf `ACCEPT` ändern, beispielsweise für die `INPUT`-Kette:

```
hugo:~# iptables -P INPUT ACCEPT
hugo:~#
```

Gegebenenfalls führen Sie die gleiche Änderung auch für die `OUTPUT`- und `FORWARD`-Ketten aus. Dann testen Sie, ob das Problem verschwunden ist. Wenn nicht, freuen Sie sich, dass Sie die Firewall-Regeln nun doch nicht genauer studieren müssen, und stellen mit

```
hugo:~# iptables -P INPUT DROP
hugo:~#
```

den alten Zustand wieder her; Gleiches gilt für die anderen Ketten, sofern Sie deren Policy verändert hatten.

Wenn das Regelwerk so kompliziert aufgebaut ist, dass eine Änderung der Policy nicht hilft (weil es eine Mischung von `ACCEPT`- und `DROP`-Regeln enthält), können Sie stattdessen einen Kurzschluss einbauen, beispielsweise für die INPUT-Kette:

```
hugo:~# iptables -I INPUT 1 -j ACCEPT
hugo:~#
```

Dadurch werden alle Datenpakete sofort akzeptiert; die weiteren Regeln bleiben unwirksam und können unverändert stehen bleiben.

Sobald die Firewall wieder normal arbeiten darf, entfernen Sie mit

```
hugo:~# iptables -D INPUT 1
hugo:~#
```

die Kurzschlussregel wieder. Gleichermaßen können Sie bei Bedarf die anderen Regelketten vorübergehend außer Gefecht setzen.

Regeln für die Address Translation lassen sich meist nicht so einfach abschalten, da sich dadurch das Verhalten der Firewall auch nach außen hin sichtbar verändern würde; beispielsweise könnten die Benutzer im internen Netz nicht mehr mit Source-NAT ins Internet geroutet werden.

Was macht die Firewall wirklich?

Um herauszubekommen, was die Firewall-Regeln mit den Datenpaketen tatsächlich anrichten, können Sie sich zweier nützlicher Hilfsmittel bedienen: Zum einen können Sie an geeigneten Stellen Logging-Regeln in die Netfilter-Regelketten einfügen, die ja keinen Einfluss auf die Behandlung der Datenpakete haben, aber durch Einträge ins Syslog dokumentieren können, welche Pakete an welcher Stelle in einer Regelkette angekommen sind.

Achtung

Verwenden Sie nach Möglichkeit die `--limit`-Option, um die Menge der entstehenden Logeinträge zu begrenzen. Wenn dennoch das Syslog zu sehr angewachsen und die `/var`- oder gar root-Partition vollgelaufen ist, könnte Ihr Syslog-Dämon seine Arbeit eingestellt haben. Damit Sie ohne Reboot weiter testen können, müssen Sie zunächst Platz auf der Festplatte schaffen und, sofern diese auf Ihrem System installiert sind, beide für das Logging zuständigen Dämonen, `syslogd` und `klogd`, neu starten.

Zum anderen können Regeln ohne Aktionsteil eingefügt werden, die ebenfalls wirkungslos sind, aber eigene Paket- und Bytezähler führen. Diese kann man abfragen, und aus ihrem Anwachsen kann man Rückschlüsse darauf ziehen, welchen Weg

die Pakete genommen haben. Diese Methode liefert weniger konkrete Informationen als die zuerst genannte, überflutet aber auch nicht das Syslog mit Einträgen und kann problemlos über längere Zeit hinweg eingesetzt werden.

Wundern Sie sich aber nicht, wenn die Zählerstände niedrig sind, obwohl viele Pakete durch die Firewall laufen: Wenn Sie, wie wir es im Firewall-Kapitel empfohlen haben, eine Firewall-Regel mit `--state`-Option benutzen und alle Pakete akzeptieren, die zu einer als `ESTABLISHED` oder `RELATED` geltenden Verbindung gehören, dann zählen die nachfolgenden Regeln nur noch die neuen Verbindungsaufnahmen!

12.3.7 Wo laufen sie denn, die Daten?

Wenn alles darauf hindeutet, dass die Infrastruktur in Ordnung ist – Verbindungen, Adressen, Routen, Firewalling – und dennoch das Netz nicht richtig funktioniert, dann bleibt noch als (nahezu) letzte Chance, um den Fehler ohne Meditation oder fremde Hilfe zu finden, die Beobachtung des nackten Datenverkehrs. Diese Möglichkeit können Sie auch dann nutzen, wenn Sie sich noch nie mit den Bits und Bytes der Internet-Protokolle beschäftigt haben und zum Einschlafen weiterhin Schäfchen zählen und nicht, wie es manchen Experten nachgesagt wird, CRC-Prüfsummen von IP-Paketen berechnen.

Es gibt zwei gängige, beliebte Programme, mit deren Hilfe man den Datenverkehr an einem Interface auf Paketebene verfolgen kann: das Kommandozeilenprogramm tcpdump und den mit einer grafischen Oberfläche versehenen wireshark (früher unter dem Namen ethereal bekannt). Beide verwenden intern die `pcap`-Bibliothek, die tief in den Niederungen des Linux-Kernels – natürlich nur mit root-Rechten – die ankommenden und abgehenden Daten aus den Puffern der Interfaces auslesen kann.

Weil bei regem Netzverkehr die dort anfallende Datenmenge sehr groß sein kann und möglicherweise von den Programmen nicht mehr schnell genug verarbeitet wird, können Filterbedingungen festgelegt werden, mit deren Hilfe die `pcap`-Routinen sehr frühzeitig entscheiden können, ob sich der Anwender überhaupt für ein bestimmtes Datenpaket interessiert. Bei sinnvoller Einrichtung des Filters kann auf diese Weise ein hohes Datenaufkommen bewältigt werden; den Programmen werden nur die für sie relevanten Daten zugestellt, und ihnen bleibt noch genügend Zeit für die weitere Auswertung.

In Anbetracht dieser gemeinsam genutzten Infrastruktur ist es also auch nicht verwunderlich, dass tcpdump und wireshark (ebenso wie einige andere, weniger populäre Programme) die gleiche, ziemlich gewöhnungsbedürftige Syntax anbieten, um die interessierenden Pakete aus dem Datenstrom einer Netzwerkschnittstelle auszufiltern. Beide Programme verwenden auch dasselbe Dateiformat zum Abspeichern von Paketen; es ist daher problemlos möglich, mit dem einen Programm die Daten abzugreifen und in eine Datei zu speichern und diese mit dem anderen Programm wieder einzulesen und auf dem Bildschirm anzeigen zu lassen.

Achtung

Normalerweise sind für die Fehlersuche nur die Anfänge eines jeden Datenpakets von Interesse; sie enthalten die Adress- und Steuerinformationen der einzelnen Protokollschichten. Deshalb werden standardmäßig nur die ersten 96 Bytes eines Pakets aufgezeichnet und interpretiert. Bei Bedarf kann aber, um eine Analyse auch auf Ebene der Anwendungsprotokolle durchführen zu können, das gesamte Paket erfasst und ausgewertet werden.

Wenn Sie ein Netzwerk administrieren, in dem nicht nur Ihre eigenen Daten, sondern auch die Daten anderer Benutzer (Kollegen, Mitbewohner, Schüler, Bundesminister) bewegt werden, bekommen Sie unter Umständen deren Kommunikation im Klartext zu sehen. Vergewissern Sie sich, unter welchen Bedingungen dies überhaupt zulässig ist und welche Verpflichtungen und juristischen Gefahren Ihnen daraus erwachsen können! WANL – *we are no lawyers*; wir werden uns hüten, Ihnen hier eine Rechtsberatung anzubieten!

tcpdump

Beim Start von `tcpdump`, der, wie gesagt, root-Rechte erfordert, können verschiedene Optionen übergeben werden. Grundsätzlich ist es sinnvoll, `-n` anzugeben, damit das Programm gar nicht erst versucht, alle Adressen und Protokollnummern in Namen umzuwandeln. Die numerische Ausgabe ist schneller und zudem übersichtlicher. Sie können IP-Adressen bei Bedarf mit dem Kommando `host` oder `nslookup`, soweit das überhaupt möglich ist, in DNS-Namen übersetzen lassen und zur Interpretation von Portnummern auf die Datei `/etc/services` zurückgreifen.

Wenn Ihr Rechner mehrere Netzwerkschnittstellen hat, sollten Sie `tcpdump` mit der Option `-i` mitteilen, welche Sie belauschen wollen. Damit wäre das Kommando schon komplett:

```
hugo:~# tcpdump -n -i ath0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ath0, link-type EN10MB (Ethernet), capture size 96 bytes
18:37:03.052066 IP 192.168.42.12.47836 > 192.168.42.6.22:
   P 1924343333:1924343413(80) ack 2239994525 win 1578
   <nop,nop,timestamp 4570858 43400902>
18:37:03.052849 IP 192.168.42.6.22 > 192.168.42.12.47836:
   . ack 80 win 16022 <nop,nop,timestamp 43412029 4570858>
18:37:04.434135 IP 192.168.42.6.1194 > 192.168.42.12.32768: UDP, length 53
18:37:04.780742 802.1d config 8000.00:0f:66:d9:99:0f.8002 root
   8000.00:0f:66:d9:99:0f pathcost 0 age 0 max 20 hello 2 fdelay 0
18:37:05.447030 IP 192.168.42.12.32768 > 192.168.42.6.1194: UDP, length 53
```

```
18:37:05.555350 IP 192.168.42.12.47836 > 192.168.42.6.22:
   P 80:192(112) ack 1 win 1578 <nop,nop,timestamp 4571484 43412029>
 STRG + C
hugo:~#
```

Wir haben die Ausgabezeilen manuell umbrochen, um sie lesbarer zu machen; im Original erscheint zu jedem Datenpaket eine Zeile, die in Abhängigkeit von der Zeilenlänge des Konsolenfensters umbrochen wird. Nachdem mehrere Pakete protokolliert worden sind, haben wir die Ausgabe mit Strg + C abgebrochen, denn sie würde sonst beliebig lange weiterlaufen.

Damit man den zeitlichen Ablauf rekonstruieren kann, beginnt jede Ausgabezeile mit einem Zeitstempel (mit einer Auflösung von einer Mikrosekunde). Bei IP-Paketen folgen die Kennzeichnung `IP` und die Absender- und Empfängeradressen (mit einem `>` als Richtungspfeil dazwischen), gefolgt von der Angabe `UDP` und der Paketlänge bei UDP-Paketen. TCP-Pakete erscheinen viel kryptischer; es werden zahlreiche Details angezeigt, die es dem Experten gestatten, die Protokollphasen und die Beziehungen zwischen aufeinanderfolgenden Paketen zu analysieren. Wenn Sie nicht gerade fließend TCP sprechen, können Sie diese Informationen getrost ignorieren.

Ohne sich mit den protokollarischen Feinheiten auseinanderzusetzen, lässt sich mit etwas Fantasie schon abzulesen, was hier vor sich geht: Zwei Stationen mit den IP-Adressen 192.168.42.6 und 192.168.42.12 unterhalten sich miteinander (aufgepasst: Als fünfte Zahl hängt an den Adressen jeweils noch die Portnummer dran!), und erfreulicherweise laufen die Pakete in beiden Richtungen. Jeweils eine der Portnummern hat einen niedrigen Wert, der darauf schließen lässt, welches Protokoll benutzt wird: `22` steht für die Secure Shell, `1194` für OpenVPN. Diese servicespezifischen Nummern stehen gewöhnlich auf der Seite des Servers; clientseitig wird eine zufällige, hohe Portnummer benutzt, die keine weitere Bedeutung hat. In unserem Beispiel lassen sich also zwei offenbar funktionierende Verbindungen identifizieren, und in beiden Fällen agiert die 192.168.42.6 als Server.

Die mit `802.1d` gekennzeichnete Nachricht gehört offensichtlich nicht zur IP-Protokollfamilie. Eine Recherche im Internet würde offenbaren, dass es sich um die Bezeichnung für ein IEEE-Protokoll handelt, mit dem Switches im Netz untereinander kommunizieren und unter alternativen Verbindungswegen die besten auswählen (auch als STP, »Spanning Tree Protocol« bekannt). Wenn Sie Ihren Datenverkehr beobachten, werden Sie noch weitere Pakete entdecken, die von den verschiedensten Geräten im Netz ausgesendet werden, um ihresgleichen oder andere reizvolle Kommunikationspartner zu finden.

Wenn Ihre erste Begeisterung über solche neu entdeckten Aktivitäten nachgelassen hat, erinnern Sie sich sicher an unsere Erwähnung der Filtermöglichkeiten, mit denen Sie die interessanteren Pakete herausdestillieren können. Die Filterregeln werden einfach an das `tcpdump`-Kommando angehängt, alle anderen Optionen müssen davor stehen. Hier sind einige einfache(!) Beispiele:

- `host 192.168.42.12`
  Alle IP-Pakete, die die angegebene Adresse als Quelle oder Ziel haben.
- `host 192.168.42.6 and host 192.168.42.12`
  Die beiden Adressen müssen als Quelle und Ziel (in beliebiger Richtung) auftauchen; es werden also alle IP-Pakete selektiert, die zu Verbindungen zwischen den beiden Rechnern gehören.
- `host 192.168.42.6 or host 192.168.42.12`
  Eine logische Oder-Verknüpfung: Mindestens eine der Adressen muss in den (IP-)Paketen als Absender oder Empfänger eingetragen sein.
- `net 192.168.42.0/24`
  Alle IP-Pakete, deren Quelle oder Ziel im angegebenen Netz liegt.
- `tcp port 22`
  TCP-Pakete, die an Port 22 gerichtet sind oder von Port 22 kommen. De facto sind dies sämtliche Pakete, die zu Secure-Shell-Verbindungen gehören.
- `udp dst port 1194`
  Hiermit werden OpenVPN-Pakete selektiert (die den UDP-Port 1194 benutzen). Das zusätzliche Schlüsselwort `dst` (für »destination«, Zielort) schränkt die Auswahl auf eine Richtung ein, bei der der Port 1194 das Ziel darstellt. Es werden also nur Pakete *zum* OpenVPN-Server gewünscht. Beachten Sie die Position des Schlüsselworts `dst`; an anderer Stelle darf es nicht stehen!
- `src host 192.168.1.12 and tcp port 631`
  Jetzt werden wir langsam übermütig: `src` (für »source«, Quelle) ist das Gegenstück zu `dst` und wählt den Host `192.168.1.12` als Absender der Pakete aus; zugleich wird verlangt, dass es sich um TCP-Pakete von oder an Port 631 handeln soll (dieser Port »gehört« IPP, dem Druckprotokoll, das auch von CUPS verwendet wird).
- `icmp[icmptype] = icmp-echo`
  Was diese Bedingung auswählen soll, ist leicht zu erraten: ICMP-Pakete, deren ICMP-Typ `icmp-echo` lautet; es handelt sich also um Pings. Die Syntax wirkt zugegebenermaßen eigenartig; sie drückt die technische Sichtweise aus, dass das Typ-Byte im ICMP-Header betrachtet und mit dem Kennwert für den Echo-Request (8) verglichen werden soll.
- `icmp[icmptype] = icmp-echo or icmp[icmptype] = icmp-echoreply`
  Sie können erraten, was dieser Ausdruck selektiert? Richtig, alle Ping- und Pong-Pakete, also Echo-Request oder Echo-Reply.

Eine genauere Erklärung sowie viele weitere Filterelemente finden Sie in der Manpage des `tcpdump`-Kommandos. Verzweifeln Sie aber nicht, wenn Ihre Versuche, eigene Filter zu definieren, immer wieder als syntaktisch falsch abgewiesen werden. Versuchen Sie, in der Manpage oder im Internet Beispiele für das zu finden, was Sie formulieren wollen, und probieren Sie mehrere Variationen durch. Bei dieser Vorgehensweise handelt es sich um einen bewährten Initiationsritus, den alle fortgeschrittenen Tcpdumpler durchmachen mussten.

Durch die Option `-w <dateiname>` können Sie verlangen, dass `tcpdump` die eingefangenen Pakete nicht auf die Standardausgabe, sondern in die angegebene Datei (eine »Capture-Datei«) schreibt[4]. Dazu wird kein lesbares Textformat, sondern eine interne Darstellung verwendet, sodass `tcpdump -r <dateiname>` die Datei wieder einlesen und die Informationen über die Datenpakete erneut verarbeiten kann. Damit lässt sich ein Mitschnitt des Datenverkehrs aufbewahren und später jederzeit wieder auswerten. Wie bereits erwähnt, ist ein Austausch solcher Dateien zwischen `tcpdump` und `wireshark` problemlos möglich.

Mit `-s 0` wird verlangt, die Pakete in voller Länge aufzuzeichnen, um Fehler auf der Anwendungsebene zu finden. Dazu kann es notwendig sein, den gesamten Dateninhalt zu analysieren.

Als letzte Option wollen wir `-e` erwähnen, die zusätzlich die Ethernet-Header der Pakete anzeigt. Normalerweise würde diese Information eher stören, aber sie kann bei der Verfolgung hartnäckiger Fehler nützlich sein, denn bisweilen sind Pakete (in sehr komplexen Netzwerkumgebungen) auf der Ethernet-Ebene an einen anderen Rechner adressiert, als die IP-Zieladresse glauben macht.

Achtung

Auch wenn Sie mit tcpdump Pakete auf einem Firewall-Rechner abgreifen, der als Gateway ein ganzes Subnetz mit dem Rest der Welt verbindet, können Sie nicht den Datenverkehr *innerhalb* des Subnetzes beobachten!

Sofern die Verbindungen innerhalb des Subnetzes über Switches hergestellt werden und diese ordnungsgemäß funktionieren, bekommt das Gateway den internen Verkehr überhaupt nicht zu sehen. (Das ist auch gut so, denn sonst würde es leicht überlastet.) Viele Switches – allerdings nicht unbedingt die billigsten – bieten die Möglichkeit, einen Anschluss als sogenannten Monitorport zu konfigurieren, dem alle Pakete in Kopie zugestellt werden, die durch den Switch wandern (soweit die Datenrate dies zulässt). Wenn Sie einen Linux-Rechner an den Monitorport anschließen, können Sie mit tcpdump oder wireshark den Verkehr mitschneiden.

Der Drahthai: wireshark

Einige Linux-Distributionen enthalten im KDE-Startmenü zwei Einträge, um wireshark zu starten: einmal mit root-Rechten (indem beim Start in einem Popup-Fenster das root-Passwort abgefragt wird) und einmal ohne. In der letzteren Variante wird das Programm nicht in der Lage sein, Daten von den Netzwerkschnittstellen abzugreifen, aber es ist gut dazu geeignet, Capture-Dateien darzustellen, die mit

[4] Es ist ratsam, während der Aufzeichnung die Dateigröße zu beobachten und das Programm zu beenden, bevor die Festplatte voll ist.

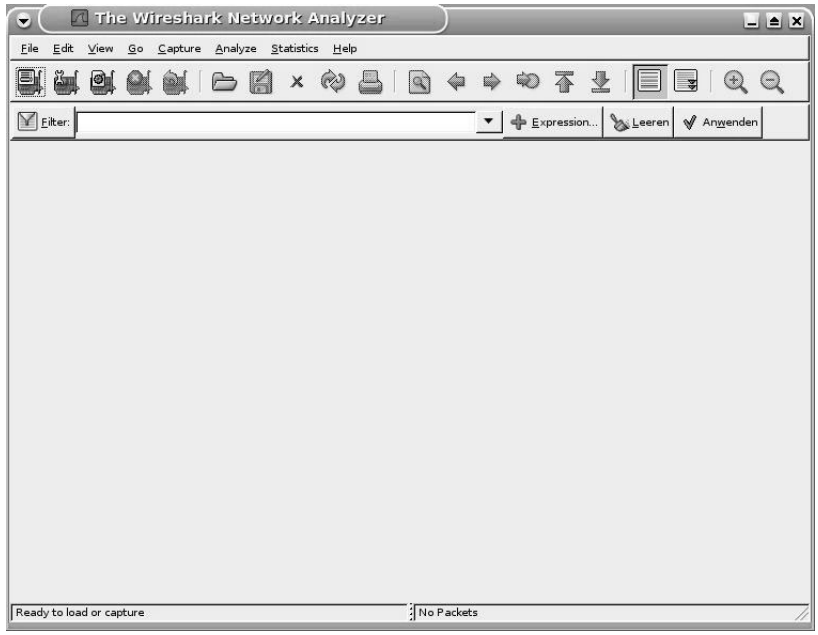


Abbildung 12.1: Startseite des Programms wireshark

tcpdump oder einem root-privilegierten wireshark – möglicherweise auf einem anderen Rechner – erstellt wurden. Hierzu können Sie den Menüeintrag FILE/OPEN[5] benutzen und im anschließenden Dateiauswahldialog die zu ladende Capture-Datei bestimmen (Abb. 12.1).

Wenn Sie hingegen mit einem unter root-Rechten laufenden wireshark Pakete aufzeichnen wollen, wählen Sie am besten den Menüeintrag CAPTURE/OPTIONS. Daraufhin erscheint ein Fenster, in dem zahlreiche Optionen verfügbar sind, die Ihnen von tcpdump her bekannt sein werden (Abb. 12.2).

Unter anderem finden Sie ein Eingabefeld, in das die Ihnen jetzt bekannten Filterausdrücke eingetragen werden können. Wenn Sie es frei lassen, schneidet das Programm alle Pakete mit, derer es habhaft werden kann.

Durch Drücken des START-Buttons wird die Aufzeichnung gestartet. Normalerweise werden nur die Anzahlen der registrierten Pakete in einem kleinen Popup-Fenster angezeigt, während die Aufzeichnung läuft. Erst nach Betätigen des STOP-Buttons erscheinen die Paketdaten im Hauptfenster des Programms. Wenn Sie aber im Optionsfenster die Checkbox UPDATE LIST OF PACKETS IN REAL TIME aktivieren, können Sie die aufgezeichneten Pakete live mitverfolgen.

[5] Die uns vorliegenden Versionen des Programms präsentierten sich nur mit englischer Oberfläche.

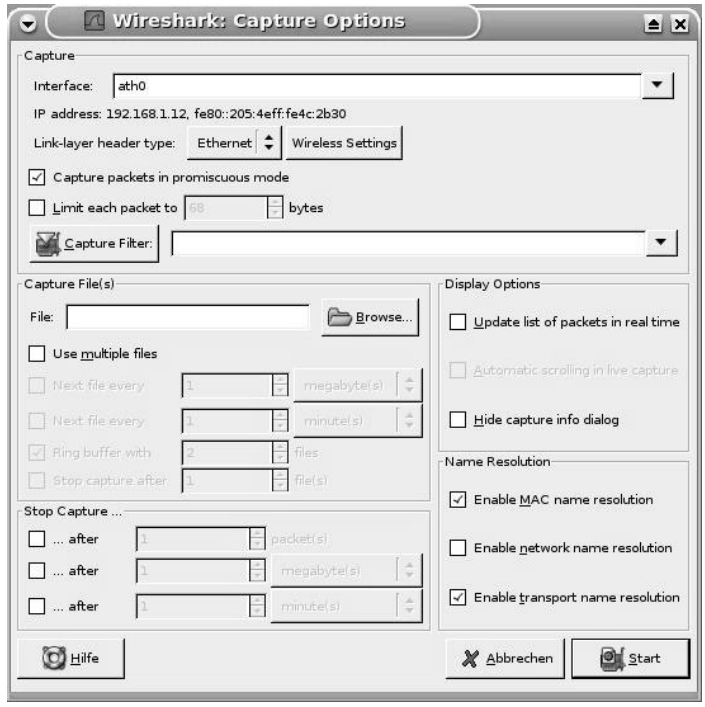


Abbildung 12.2: wireshark-Optionen

Anschließend zeigt Ihnen der zentrale Teil des Hauptfensters (Abb. 12.3) die Paketliste, deren Einträge farbig gekennzeichnet sind. Dadurch werden verschiedene Pakettypen unterschieden; besonders auffällig (in Rot/Schwarz) werden Pakete eingefärbt, die auf mögliche Probleme hinweisen, beispielsweise wiederholte Pakete (»retransmissions«), TCP-Reset-Pakete oder solche, die nicht in den aufgezeichneten Datenfluss passen. Wenn Sie also auf Fehlersuche sind, sollten Sie diese Warnmarkierungen dankbar annehmen und genauer in Augenschein nehmen – vielleicht weisen sie auf die wahre Ursache der Probleme hin.

Unterhalb der Paketliste findet sich ein kleinerer Bereich, in dem für ein ausgewähltes Paket die Datenstruktur nach Protokollebenen säuberlich getrennt angezeigt wird, die niedrigste Ebene zuoberst. Durch Anklicken des kleinen Dreiecks am linken Rand können Sie jede der Ebenen »öffnen«; wireshark zeigt dann sämtliche Details des jeweiligen Protokolls an. Im darunterliegenden Bereich des Fensters werden die rohen Daten des Pakets hexadezimal angezeigt, wobei eine selektierte Protokollebene zusätzlich gekennzeichnet ist.

Der Aufruf von wireshark und das Studium einer Reihe aufgezeichneter Pakete ist auch dann zu empfehlen, wenn Sie überhaupt keinen Fehler zu suchen haben – man kann eine Menge über die Internet-Protokollsuite lernen!

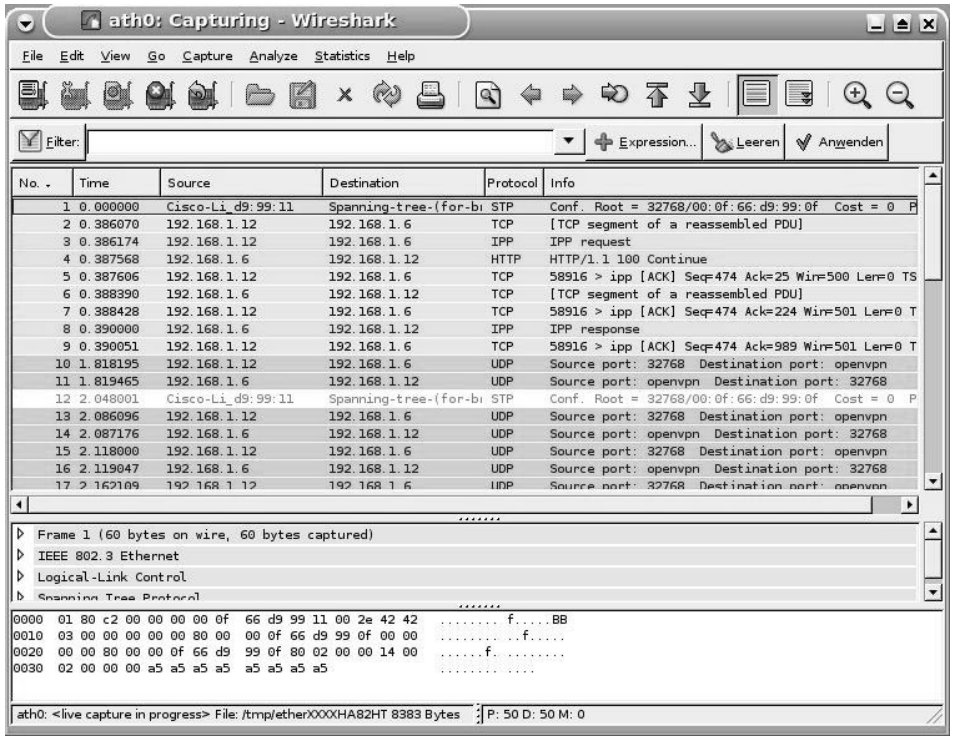


Abbildung 12.3: wireshark: Hauptfenster mit Paketliste

Die guten Analysefunktionen dieses Programms bergen allerdings auch Risiken; es ist immer mal wieder vorgekommen, dass die Routinen zur protokollspezifischen Auswertung Programmierfehler enthielten, die es (zumindest theoretisch) einem Angreifer ermöglicht hätten, das Programm (das ja oft unter der root-Kennung läuft) unter seine Kontrolle zu bringen. Gegen solche Angriffe kann man sich weitestgehend schützen, indem man den Netzverkehr von tcpdump aufzeichnen lässt und anschließend auf einem gut gesicherten, vielleicht nicht einmal vernetzten Rechner unter einer normalen Benutzerkennung auswertet.

Anhang

A.1 Hintergründiges: Blicke hinter die Kulissen

Wir wollen mit diesem Buch kein Grundlagenwerk schaffen, weder zu den Internetprotokollen im Allgemeinen noch zu deren Linux-Implementation im Besonderen. Andererseits glauben wir, dass es beim Einsatz von Linux helfen kann, eine grundlegende Vorstellung davon zu haben, was unter der bunten grafischen Oberfläche passiert, wenn Netzwerke konfiguriert und vernetzte Anwendungen benutzt werden.

Perfekte Kenntnisse des Linux-Networking sind keineswegs erforderlich, um das System zu nutzen; die können wir nicht vorweisen, und wenn man es wirklich ganz genau wissen will, kann ohnehin nur der Blick in den Quellcode offenbaren, was tatsächlich unter welchen Bedingungen wie funktioniert, denn auch die Dokumentation ist immer ungenau und lückenhaft. Eine modellhafte Vorstellung der wichtigsten Abläufe im Netz und im Linux-Kernel kann aber sehr hilfreich sein, um ein Netzwerk zu planen und Fehlern im Betrieb auf die Spur zu kommen.

Deshalb möchten wir Ihnen im Folgenden einige grundlegende Zusammenhänge skizzieren, ohne Anspruch auf Vollständigkeit und mathematische Exaktheit zu erheben. Damit Sie die vorhergehenden Kapitel besser durchstehen konnten (was Ihnen, wie wir hoffen, gelungen ist, wenn Sie uns bis hierhin noch lesend die Treue halten), haben wir diesen etwas trockeneren Stoff in den Anhang verbannt.

A.1.1 IP-Adresse und Netzmaske

IP-Adressen[1] werden fast immer als »dotted quad«, als Folge von vier Zahlen geschrieben, die durch Punkte voneinander getrennt sind. Jede dieser Zahlen steht für ein Byte der Adresse und kann folglich Werte von 0 bis 255 annehmen. Dies gilt jedenfalls für die derzeit nahezu ausschließlich benutzte Version 4 des IP-Standards, auch kurz IPv4 genannt. Die Notation für die seit Jahren kurz vor der allgemeinen Einführung stehende, 128 Bit lange Adressen umfassende Version des Internet-Protokolls (IPv6) ist wesentlich komplizierter und begegnet Ihnen bereits in einigen Ausgaben von Linux-Kommandos; da IPv6 noch immer keine nennenswerte Rolle spielt, wollen wir in diesem Buch (zumindest in den ersten neun Auflagen …) nicht näher darauf eingehen.

[1] Bei sehr oft benutzten Abkürzungen vergisst man leicht, wofür sie überhaupt stehen – IP bedeutet »Internet Protocol«.

Ursprünglich konnten nur solche Adressen, die in den ersten ein, zwei oder drei Bytes übereinstimmen, zu Subnetzen zusammengefasst werden (sogenannte Klasse-A-, B- bzw. C-Netze). Mittlerweile kann man es sich nicht mehr leisten, z. B. für jedes Subnetz, das mit 256 Adressen (Klasse C) nicht auskommt (von denen nur 254 tatsächlich nutzbar sind), gleich $256 * 256 = 65.536$ Adressen zu verplanen; deshalb kann die Anzahl der Bits, die bei allen Adressen eines Subnetzes übereinstimmen muss, flexibel gewählt werden.

Statt von Klasse-C-Netzen sollte man also heutzutage von 24-Bit-Subnetzen sprechen. In der CIDR-Notation (»Classless Inter-Domain Routing«, klassenloses Routing) schreibt man dies beispielsweise in der Form 192.168.42.0/24. Die 24 wird dabei als Präfixlänge bezeichnet.

Ein solches Subnetz umfasst alle Adressen von 192.168.42.0 bis 192.168.42.255. Etwas mehr Kopfrechnen ist erforderlich, um aus »krummen« Subnetzangaben wie 192.168.42.64/26 die zugehörigen Adressen abzulesen:

```
Subnetzadresse   192.168.42.64 = binär 11000000 10101000 00101010 01000000
26 Bit lange Netzmaske:          binär 11111111 11111111 11111111 11000000
kleinste Adresse 192.168.42.64 = binär 11000000 10101000 00101010 01000000
höchste Adresse 192.168.42.127 = binär 11000000 10101000 00101010 01111111
```

Zu diesem Subnetz gehören also die Adressen 192.168.42.64 bis 192.168.42.127; wirklich nutzbar als Stationsadressen sind von diesen alle außer der ersten und der letzten, also die 62 Adressen von 192.168.42.65 bis 192.168.42.126.

Die Folge von (hier 26) Eins-Bits in der Länge der Subnetzangabe wird als Netzmaske bezeichnet und oft auch – etwas umständlicher – in der Dotted-Quad-Notation geschrieben, in diesem Beispiel als 255.255.255.192.

A.1.2 Wie findet ein IP-Paket den Weg zum Empfänger?

Es gibt zwei Möglichkeiten – einen direkten und einen indirekten Weg. Der direkte Weg wird gewählt, wenn sich Sender und Empfänger im selben IP-Subnetz befinden. In diesem Fall trägt der Absender die MAC-(Ethernet-)Adresse des Empfängers in das Paket ein, und das Fußvolk der IP-Infrastruktur vom Kabel bis zum Switch geleitet die Daten von der sendenden bis zur empfangenden Netzwerkkarte.

Dazu muss der Absender aber zunächst einmal wissen, ob diese direkte Zustellung überhaupt möglich ist. Er stellt das einfach anhand der IP-Adresse fest, indem er diese mit seiner eigenen Adresse vergleicht. Wenn von links aus gerechnet so viele Bits beider Adressen übereinstimmen, wie durch die zum Subnetz gehörende Netzmaske vorgegeben ist, kann die Zieladresse direkt erreicht werden.

Ein Beispiel hierzu:

```
1. IP-Adresse: dezimal 192.168.42.2, binär 11000000 10101000 00101010 00000010
   24-Bit-Netzmaske 255.255.255.0,   binär 11111111 11111111 11111111 00000000
2. IP-Adresse: dezimal 192.168.42.6, binär 11000000 10101000 00101010 00000110
```

Die beiden IP-Adressen 192.168.42.2 und 192.168.42.6 stimmen in allen Bits, die durch Einsen in der Netzmaske ausgewählt sind, überein und gehören somit zum selben Subnetz.

Um sich nun mit einem benachbarten Rechner innerhalb des Subnetzes in Verbindung zu setzen, braucht der Absender dessen MAC-Adresse. Früher hat man die Zuordnung von MAC- und IP-Adressen einfach in eine Datei (`/etc/ethers`) eingetragen:

```
00:10:24:27:21:21 192.168.42.1
00:10:24:27:2c:24 192.168.42.2
00:18:35:c9:1a:47 192.168.42.6
```

Sobald ein Netz größer wird oder hin und wieder mal Veränderungen erfährt, ist die Pflege einer solchen Datei natürlich höchst unpraktisch. Deshalb gibt es zur automatischen Adressauflösung ARP, das Address Resolution Protocol: Der Absender schickt eine Ethernet-Nachricht als Broadcast, also als Rundruf, ins Netz hinaus, der allen Stationen im Subnetz zugestellt wird und die Frage enthält: »Wer von euch hat die IP-Adresse 192.168.42.2? Viele Grüße, Euer 192.168.42.6.« Im Datenpaket ist die Anfrage viel kompakter, aber inhaltsgleich formuliert; ein Schnüffelprogramm wie wireshark (früher unter dem Namen ethereal bekannt) oder tcpdump zeigt sie uns lesbar an:

```
hugo:~# tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
13:36:31.454164 arp who-has 192.168.42.2 tell 192.168.42.6
13:36:31.455030 arp reply 192.168.42.2 is-at 00:0f:66:d9:99:0f
 STRG + C
```

Im günstigen Fall erhält also der Fragesteller eine Antwort mit der passenden MAC-Adresse und kann an diese sofort die Daten schicken. Bekommt er keine Antwort, versucht er es noch ein paarmal und gibt schließlich auf. Der Zielrechner ist dann vermutlich ausgeschaltet, schlecht gelaunt oder existiert überhaupt nicht.

Ergibt der Adressenvergleich, dass das Ziel nicht im selben Subnetz liegt, so muss der indirekte Weg beschritten werden, und der führt über einen Router. Die meisten Rechner im Netz kennen genau einen solchen, den sogenannten Defaultrouter, dem sie alle Pakete anvertrauen, die sie im eigenen Subnetz nicht loswerden können. Dieser Router stellt das Tor (das Gateway) zur weiten Welt dar und muss im lokalen Subnetz erreichbar sein. Seine MAC-Adresse kann also per ARP ermittelt werden. Wenn nicht, hat der Absender verloren – sämtliche fremden Netze sind dann unzugänglich. Die Datenpakete werden alsdann mit der IP-Adresse des Zielrechners, aber der MAC-Adresse des Routers auf die Reise geschickt, und wenn der Router seinem Namen Ehre macht, nimmt er die Pakete entgegen, ermittelt anhand der Zieladresse einen Weg, der zu dieser hinführt (oft über einen oder mehrere

weitere Router), und sendet die Daten weiter. Dabei behält das Paket im Normalfall seine ursprünglichen Absender- und Ziel-IP-Adressen, reist aber mit neuen MAC-Adressen weiter: der des Routers als Quelle, der der nächsten Zwischenstation (oder des Empfängers) als Ziel.

Dadurch, dass die Pakete sowohl auf Ethernet- als auch auf IP-Ebene die jeweiligen Quell- und Zieladressen enthalten, finden auch durch Vertauschen dieser Adressen die Antwortpakete ihren Weg durchs Netz.

Um sich also mit seinesgleichen – nah oder fern – unterhalten zu können, benötigt ein Rechner (egal, ob er unter Linux läuft oder nicht):

- eine MAC-Adresse (die jede Netzwerkkarte als weltweit eindeutigen Wert schon ab Werk mitbringt),
- eine IP-Adresse,
- die Netzmaske des lokalen Subnetzes, zu dem er gehört, und
- die Adresse des Defaultrouters.

Festgelegt werden diese Werte (außer der MAC-Adresse) normalerweise bei der Einrichtung der Netzwerkkarte; abfragen lassen sich die ersten drei mit dem Kommando `ifconfig` oder, Linux-spezifisch und moderner, mit `ip addr show`, die letztere mit `route -n` bzw. `ip route show`.

Alle mal herhören – Broadcasts

Nicht so zwingend erforderlich, aber in der Regel ratsam ist die Zuweisung einer subnetzspezifischen Broadcast-Adresse, die üblicherweise als höchste Adresse des Subnetzes vereinbart wird, d. h., aus der IP-Adresse abgeleitet wird, indem man alle von der Netzmaske nicht erfassten Bits der Adresse auf 1 setzt. Lautet die IP-Adresse beispielsweise 192.168.42.7 und umfasst die Netzmaske 24 Bit, so ergibt sich als Broadcast-Adresse 192.168.42.255.

Die Vereinbarung einer Broadcast-Adresse ermöglicht es, auf der IP-Ebene eine Nachricht an alle anderen Geräte im selben Subnetz zu senden, und wenn man Glück hat, fühlen diese sich auch alle angesprochen. Sendet zum Beispiel ein Programm eine UDP-Nachricht an die Broadcast-Adresse, so versucht das Betriebssystem gar nicht erst, den Empfänger per ARP zu ermitteln, sondern schickt das Datenpaket mit der Ziel-MAC-Adresse ff:ff:ff:ff:ff:ff auf die Reise. Dadurch fühlen sich alle direkt erreichbaren Stationen angesprochen, und wenn sie die Broadcast-IP-Adresse als Ziel der Nachricht erkennen, verwerfen sie das Paket nicht sofort, wie sie es sonst normalerweise tun, wenn als Ziel nicht die eigene IP-Adresse eingetragen ist.

Es gibt einige Anwendungsprotokolle, die auf diese Weise versuchen, Kontakt zu bisher unbekannten Partnern aufzunehmen. CUPS beispielsweise kann sich der Broadcasts bedienen, um alle Clients im Subnetz über die zur Verfügung stehenden Drucker zu informieren.

A.1.3 Die Kommandos ifconfig und ip

Sie werden sich vielleicht wundern, dass Sie in verschiedenen Anleitungen zur Linux-Netzwerkkonfiguration – und auch in diesem Buch – unterschiedlichen Kommandos begegnen, mit denen die Schnittstellen und auch das Routing eingerichtet werden. Tatsächlich bietet Linux zwei unterschiedliche Kommandowelten für die netzwerkbezogenen Einstellungen an: Zum einen die klassischen Unix-Kommandos `ifconfig`, `route` und `arp`, zum anderen das universelle, Linux-spezifische Kommando `ip`. Letzteres bildet die kernelinternen Einstellungen ziemlich umfänglich, vielseitig, aber zugegebenermaßen gewöhnungsbedürftig auf die Kommandoschnittstelle ab. Es deckt nicht nur die Funktionalität der anderen Kommandos ab, sondern bietet viel weiter gehende Möglichkeiten. Für die »alten Hasen« werden die gewohnten Kommandos weiterhin unterstützt und erhalten ein Mindestmaß an Kompatibilität zu BSD- und anderen Unix-Systemen aufrecht, intern aber werden `ifconfig` & Co. auf die durch `ip` bereitgestellte Funktionalität abgebildet.

Deshalb ist es im Grunde gleichgültig, welche Kommandos Sie bevorzugen; Sie brauchen sich nicht einmal für eine der Welten zu entscheiden, sondern können sich jeweils desjenigen Kommandos bedienen, das Sie besser kennen oder das Ihnen besser gefällt. Lediglich aus Konsistenzgründen ist es ratsam, in Prozeduren eine gewisse Einheitlichkeit zu wahren, und wir würden – sofern Portabilität zu anderen Betriebssystemen keine Rolle spielt – die Benutzung von `ip` empfehlen, weil es mehr Möglichkeiten bietet und weniger verdeckte Nebeneffekte hat.

ifconfig versus ip

Das Kommando `ifconfig` kann eine Netzwerkschnittstelle aktivieren und deaktivieren und ihr gleichzeitig eine IP-Adresse und Netzmaske zuordnen. Dies geschieht oft im selben Aufruf:

```
ifconfig eth0 192.168.42.7 netmask 255.255.255.0 broadcast 192.168.42.255
```

aktiviert die Schnittstelle `eth0`, wenn nötig, und weist ihr eine Adresse zu. Die Angabe der Netzmaske in CIDR-Schreibweise ist unter Linux auch möglich:

```
ifconfig eth0 192.168.42.7/24 broadcast 192.168.42.255
```

Will man `eth0` nur ansprechbar machen, aber einstweilen keine Adresse konfigurieren, muss man zu einem Trick greifen und die Nulladresse angeben:

```
ifconfig eth0 0.0.0.0
```

Die Deaktivierung erreicht man mit dem Zusatz `down`:

```
ifconfig eth0 down
```

Beim `ip`-Kommando sind diese Funktionalitäten strikt voneinander getrennt. Mit

```
ip link set eth0 up
```

wird `eth0` aktiviert, und erst dann lässt sich durch

```
ip addr add 192.168.42.7/24 brd + dev eth0
```

die Adresse zuweisen. Der Zusatz `brd +` gibt an, dass die Broadcast-Adresse nach der üblichen Konvention gebildet wird und die höchste Adresse innerhalb des durch die Netzmaske angegebenen Bereichs darstellt[2].

Um der Schnittstelle die Adresse wieder wegzunehmen, wird `del` statt `add` benutzt:

```
ip addr del 192.168.42.7/24 dev eth0
```

Auf die Angabe der Netzmaske kann dabei nicht verzichtet werden, ebenso wenig auf die Nennung der Schnittstelle! Schließlich versetzt das Kommando

```
ip link set eth0 down
```

das Interface wieder in den inaktiven Zustand.

Bequem und verbreitet ist die Abfrage aller Netzwerkschnittstellen mit einem einfachen `ifconfig` ohne weitere Optionen:

```
victor@hugo:~$ ifconfig
eth0      Protokoll:Ethernet  Hardware Adresse 00:0D:88:3A:55:8B
          inet Adresse:192.168.42.7  Bcast:192.168.42.255  Maske:255.255.255.0
          inet6 Adresse: fe80::20d:88ff:fe3a:558b/64 Gültigkeitsbereich:Verbindung
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:435 errors:0 dropped:0 overruns:0 frame:0
          TX packets:87 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX bytes:29485 (28.7 KiB)  TX bytes:5145 (5.0 KiB)
          Interrupt:10 Basisadresse:0xb800

lo        Protokoll:Lokale Schleife
          inet Adresse:127.0.0.1  Maske:255.0.0.0
          inet6 Adresse: ::1/128 Gültigkeitsbereich:Maschine
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:248 errors:0 dropped:0 overruns:0 frame:0
          TX packets:248 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:0
          RX bytes:18632 (18.1 KiB)  TX bytes:18632 (18.1 KiB)
```

Hingegen zeigt `ip link show` nur den Linkstatus, `ip addr show` zusätzlich die konfigurierten Adressen aller Interfaces in ziemlich kompakter Form an:

[2] Es ist auch möglich, mit `brd -` die kleinstmögliche Adresse für Broadcasts zu benutzen, hier also 192.168.42.0, aber das ist sehr unüblich, und deshalb sollten Sie es besser vermeiden. Benutzen Sie aber diese niedrigste (»network address« genannte) Adresse auch nicht als Stationsadresse!

```
victor@hugo:~$ ip addr show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0d:88:3a:55:8b brd ff:ff:ff:ff:ff:ff
    inet 192.168.42.7/24 brd 192.168.42.255 scope global eth0
    inet6 fe80::20d:88ff:fe3a:558b/64 scope link
       valid_lft forever preferred_lft forever
victor@hugo:~$ ip link show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0d:88:3a:55:8b brd ff:ff:ff:ff:ff:ff
victor@hugo:~$
```

Über die weiteren, recht zahlreichen Optionen beider Kommandos informieren die zugehörigen Manpages. Während `ifconfig` in jedem Linux-System installiert sein sollte, ist `ip` meist Bestandteil eines separaten Pakets, das üblicherweise `iproute` oder `iproute2` genannt wird. In diesem Paket sollte sich auch eine druckbare Datei mit Namen `ip-cref.ps` oder `ip-cref.ps.gz` finden, die eine ausführliche, aber nicht gerade leicht verständliche Beschreibung des `ip`-Kommandos und seiner Optionen enthält.

Navigationshelfer: route, ip route

Anhand der Routing-Tabelle bestimmt das Betriebssystem, auf welchem Weg Nachrichten an ihr Ziel zu schicken sind. Dazu versucht es, in der Tabelle den am besten zur Zieladresse passenden Eintrag zu finden; kommen mehrere Einträge in Frage, gewinnt der mit der längsten Netzmaske. Diese Routing-Entscheidung kann im Wesentlichen zu vier unterschiedlichen Ergebnissen führen:

- Die Zieladresse gehört zum lokalen Rechner, die Daten sind also direkt einer Anwendung zuzustellen.
- Die Zieladresse müsste in dem Subnetz zu finden sein, das an einer der Netzwerkschnittstellen hängt. Im Regelfall wird ARP benutzt, um die physikalische Adresse des Ziels zu ermitteln.
- Das Ziel muss über ein Gateway angesteuert werden.
- Es gibt keine Route zum Ziel, die Zustellung der Nachricht ist daher gar nicht möglich.

Die Routing-Tabelle kann mit dem Kommando `ip route show` angezeigt werden:

```
victor@hugo:~$ ip route show
192.168.42.0/24 dev eth0  proto kernel  scope link  src 192.168.42.7
```

```
default via 192.168.42.1 dev eth0
victor@hugo:~$
```

Das Beispiel zeigt eine ganz einfache Konstellation: Alle Ziele im Subnetz `192.168.42.0/24` sind über die Schnittstelle `eth0` zu routen, alles andere (dafür steht »default«) unter Mithilfe des Gateways 192.168.42.1, das ebenfalls über `eth0` zu finden ist. Bei der ersten Route ist zusätzlich vermerkt, welche Absenderadresse verwendet würde.

Das äquivalente Unix-Standardkommando lautet `route -nv`:

```
victor@hugo:~$ route -nv
Kernel IP Routentabelle
Ziel          Router        Genmask        Flags  Metric  Ref   Use Iface
192.168.42.0  0.0.0.0       255.255.255.0  U      0       0       0 eth0
0.0.0.0       192.168.42.1  0.0.0.0        UG     0       0       0 eth0
victor@hugo:~$
```

Diese Ausgabe sieht zwar völlig anders aus, stellt aber nur eine andere Sicht auf dieselben Routing-Einträge dar: Netze sind durch Adresse und Netzmaske (Genmask) beschrieben, und das Ziel `0.0.0.0` steht für den Default-Eintrag. Wenn in der Spalte `Router` eine Adresse ungleich `0.0.0.0` erscheint, so handelt es sich um das Gateway.

Beide Kommandos, `route` und `ip route`, bieten eine Reihe von Möglichkeiten, die Routing-Tabelle[3] zu manipulieren, auf die wir hier jedoch nicht näher eingehen wollen. Lediglich auf eine sehr nützliche Funktionalität des `ip`-Kommandos wollen wir hinweisen: Anstatt nämlich die Routing-Einträge selbst mühsam zu interpretieren, kann man für eine beliebige Zieladresse die Routing-Entscheidung ausführen und das Ergebnis anzeigen lassen:

```
victor@hugo:~$ ip route get 192.168.42.44
192.168.42.44 dev eth0  src 192.168.42.7
    cache  mtu 1500 advmss 1460 hoplimit 64
victor@hugo:~$ ip route get 1.2.3.4
1.2.3.4 via 192.168.42.1 dev eth0  src 192.168.42.7
    cache  mtu 1500 advmss 1460 hoplimit 64
victor@hugo:~$ ip route get 192.168.42.7
local 192.168.42.7 dev lo  src 192.168.42.7
    cache <local>  mtu 16436 advmss 16396 hoplimit 64
victor@hugo:~$
```

Beim ersten Mal wird angezeigt, dass die Route ins lokale Netz weist (`dev eth0`), beim zweiten Mal wird das Gateway eingeschaltet (`via 192.168.42.1`). Das dritte Komman-

[3] Genau genommen gibt es sogar mehrere Routing-Tabellen in einem Linux-System, die benutzt werden können, um komplizierte Routen in Abhängigkeit von Adressen und Daten zu definieren oder, was wesentlich leichter ist, sich ins Knie zu schießen.

do fragt ganz scheinheilig nach der Adresse des eigenen Rechners, und die Ausgabe weist auf eine lokale Zustellung hin (`dev lo`).

Nachbarschaft: arp, ip neigh

Damit nicht für jedes ins lokale Subnetz zu sendende Datenpaket von neuem die MAC-Adresse der Gegenstelle erfragt werden muss, merkt sich der Linux-Kernel die Ergebnisse von ARP-Anfragen in einer Tabelle. Auch negative Ergebnisse werden dort eingetragen. Allerdings dürfen alle Einträge nur eine begrenzte Lebensdauer haben, denn sonst würde das Verschwinden (oder Ausschalten) eines Nachbarrechners nicht bemerkt, ihm würden weiter Pakete zugesandt, und der Anwender bekäme nicht mit, dass die Daten im Nirwana landen.

Der Inhalt dieser ARP-Tabelle kann mit den Kommandos `arp` und `ip neigh` (*neigh* steht für *neighbour*, das englische Wort für Nachbar) angezeigt und manipuliert werden. Letzteres wollen wir hier außer Betracht lassen; im Allgemeinen ist es besser, die Veränderung der Tabelle dem Kernel zu überlassen und nur in Problemfällen selbst einzugreifen.

Wenn Sie vor kurzem Daten über das Gateway in die weite Welt gesendet haben, wird zumindest dieses in der Liste der bekannten Nachbarn auftauchen:

```
victor@hugo:~$ ip neigh show
192.168.42.1 dev eth0 lladr 00:0f:66:d9:99:4f REACHABLE
victor@hugo:~$
```

Daraus ist die Zuordnung der IP- zur MAC-Adresse (`lladr`, link layer address) zu erkennen sowie die Schnittstelle (`eth0`), an der der Nachbarrechner zuletzt gesehen wurde. Solange als Status `REACHABLE` eingetragen ist, ist der Eintrag unmittelbar verwendbar; nach einer gewissen Zeit wird sich der Status in `STALE` (»schal«) ändern, und der Kernel wird versuchen, bei Bedarf die Information aufzufrischen. Noch etwas später verschwindet der Eintrag komplett, der Zielrechner gilt als unbekannt und muss erst (per ARP) neu geortet werden.

Wenn im ARP-Tabelleneintrag die MAC-Adresse fehlt und als Status `FAILED` oder `INCOMPLETE` eingetragen ist, hat die Kontaktaufnahme nicht geklappt, und die betreffende IP-Adresse gilt als derzeit unerreichbar.

Auch hier gibt es wieder ein Kompatibilitätskommando, das die gleiche systeminterne Information etwas anders aufbereitet. Es lautet `arp`, und zur Abfrage des Status gibt man zweckmäßigerweise die Option `-n` an, die für eine numerische Ausgabe der Adressen sorgt.

```
victor@hugo:~$ arp -n
Address          HWtype  HWaddress          Flags  Mask      Iface
192.168.42.1     ether   00:0f:66:d9:99:4f  C                eth0
192.168.42.23            (unvollständig)                     eth0
victor@hugo:~$
```

Ein C in der Spalte Flags steht für »complete« (vollständig) und zeigt die Gültigkeit des Eintrags an. Der zweite Eintrag enthält keine Hardware-Adresse, seine IP-Adresse ist also derzeit nicht erreichbar.

A.1.4 Netzwerkschnittstellen

Die IP-Adresse und die Netzmaske sind, wie im Abschnitt A.1.2 beschrieben, die Grundvoraussetzungen, um überhaupt in einem Subnetz zu kommunizieren (und zusätzlich noch eine Gateway-Adresse, um aus dem Subnetz hinauszukommen). Ehe wir diese beiden Werte einer Netzwerkschnittstelle zuordnen können, müssen wir erst einmal eine solche haben. Wir wollen uns an dieser Stelle nicht damit beschäftigen, wie eine Netzwerkkarte physikalisch in Ihren Rechner kommt – notfalls müssten Sie ein bisschen an Ihrem PC herumschrauben –, sondern woher die softwaremäßige Schnittstelle kommt, die zum Vernetzen notwendig ist.

Interfaces zum Leben erwecken

Wenn beispielsweise ein USB-Netzwerk-Adapter an Ihren Linux-Rechner angestöpselt wird, für den ein Treiber im System vorhanden ist (glücklicherweise ist das meist der Fall), dann lädt der Hotplugging-Code des Systems den Treiber als Kernelmodul, das daraufhin versucht, den Adapter zu initialisieren. Wenn alles gutgeht, meldet es seine Dienste als Netzwerkschnittstelle beim Betriebssystem an und erhält als solche einen Namen, meist den ersten freien aus der Reihe `eth0`, `eth1`, ..., und kann fürderhin mit den Kommandos `ip` und `ifconfig` angesprochen werden.

Eine fest eingebaute Netzwerkkarte hingegen löst mangels Einstöpselvorgangs kein richtiges Hotplugging aus, aber der Kernel simuliert dies beim Booten. Wohl weil das System zu diesem Zeitpunkt noch nicht richtig warmgelaufen ist, bezeichnet man den Vorgang als Coldplugging. Der Effekt ist der gleiche: Wenn es ein passendes Treibermodul gibt und dies nicht in den Kernel fest einkompiliert ist, wird es geladen, und in jedem Fall erhält es die Chance, das Interface zu initialisieren und beim Kernel anzumelden. Das gilt für klassische Ethernet-Karten genauso wie für WLAN-Karten; besonders die letzteren melden sich oft nicht als `ethX` an, sondern beispielsweise als `athX` (wenn sie vom Madwifi-Treiber versorgt werden) oder als `wlanX`, wobei `X` für die jeweils erste freie Nummer (0, 1, ...) steht.

Die Vergabe der Nummern nach dem Prinzip »Wer zuerst kommt, mahlt zuerst« kann Probleme verursachen, wenn sich aus irgendeinem Grund die Reihenfolge der Initialisierung ändert: Plötzlich hängt an der Schnittstelle `eth0`, die bisher die Verbindung zum lokalen Netz darstellte, das Kabel in Richtung Internet und umgekehrt. Damit das nicht passiert, versuchen die Distributionen meist (auf unterschiedliche Weise), bei der erstmaligen Konfiguration einer Netzwerkkarte deren eindeutige MAC-Adresse mit einem festen Schnittstellennamen zu assoziieren. Dies kann beispielsweise (und nach unserem Geschmack am besten) über die Konfigurationsdateien von `udev` geschehen, des Programms, das von neuen Kernels beim Hotplugging eingeschaltet wird, um Gerätenamen und -attribute zu verwalten.

Sollten Sie versuchen, eine Schnittstelle zu konfigurieren, die nicht vorhanden ist, so tun `ip` und `ifconfig` ihr Bestes, um doch noch ein passendes Interface herbeizuzaubern: Sie versuchen, ein Kernelmodul mit dem Namen der fehlenden Schnittstelle zu laden. Nun heißt kein vernünftiges Modul `eth0` oder so ähnlich, weshalb der Versuch nur gelingt, wenn durch einen Alias-Eintrag in der Konfiguration für das `modprobe`-Kommando (z. B. im Verzeichnis `/etc/modprobe.d/`) der Schnittstellenname auf den Modulnamen abgebildet wird. Diese Methode kann bei Hardware-Interfaces hilfreich sein, die nicht richtig durch Cold- oder Hotplugging erkannt werden, oder wenn die dynamische Geräteerkennung überhaupt nicht installiert ist. Notwendig ist sie vor allem für diejenigen Interfaces, die gar nicht zu pluggen sind, weil sie nur als Software existieren. Beispiele dafür sind die Dummy-Interfaces (die in Wahrheit gar keine Daten empfangen oder verschicken können, aber sich ansonsten wie Netzwerkkarten verhalten) und die Bonding-Interfaces, die mehrere echte Schnittstellen bündeln können, um Lastverteilung oder Ausfallsicherheit zu erreichen[4].

Wenn Netzwerkschnittstellen ihr Treibermodul per Modul-Alias finden sollen, müssen auch die gegebenenfalls notwendigen Treiberoptionen in der Modprobe-Konfiguration als `options`-Einträge hinterlegt werden. Insbesondere ist für die softwareemulierten Schnittstellen schon beim Laden des Moduls, egal welcher Aliasname benutzt wird, die benötigte Anzahl festzulegen, denn geladen wird das Modul natürlich nur einmal. Für die Dummies können die Modprobe-Parameter etwa so aussehen:

```
alias dummy0 dummy
alias dummy1 dummy
alias dummy2 dummy
options dummy numdummies=3
```

Mit diesen Optionen meldet sich der Dummy-Treiber gleich dreimal beim System an. Danach können ein oder mehrere dieser Pseudointerfaces aktiviert werden.

```
hugo:~# ip link set dummy0 up
hugo:~# ip link show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: dummy0: <BROADCAST,NOARP,UP> mtu 1500 qdisc noqueue
    link/ether a2:60:b2:28:b2:c5 brd ff:ff:ff:ff:ff:ff
3: dummy1: <BROADCAST,NOARP> mtu 1500 qdisc noop
    link/ether 7a:58:b3:54:e6:bf brd ff:ff:ff:ff:ff:ff
4: dummy2: <BROADCAST,NOARP> mtu 1500 qdisc noop
    link/ether 2e:e4:09:5f:10:6d brd ff:ff:ff:ff:ff:ff
hugo:~#
```

[4] Wenn Sie meinen, dass Sie so etwas gebrauchen könnten, sollten Sie das Kommando `ifenslave` installieren, das oft in einem gleichnamigen Paket vorliegt.

Die Ethernet-Adressen der Dummies werden übrigens ausgewürfelt und so bestimmt, dass sich konventionsgemäß kein Konflikt mit echten Netzwerkkarten ergibt.

A.1.5 Was nützen Interfaces ohne Adresse?

Wir haben erwähnt, dass ein Interface nach `ip link set ... up` (oder einem `ifconfig` mit Adresse 0.0.0.0) aktiviert ist und eine IP-Adresse erhalten kann. Völlig nutzlos ist es aber auch ohne Adresse nicht; auf unterster Ebene (dem Link Layer) kann man durchaus schon (Ethernet-)Pakete senden und empfangen. Das genügt, um mit tcpdump den Datenverkehr, den das Interface empfängt, abzuhören; allerdings werden im Regelfall nur Broadcast-Pakete erscheinen, weil die eigene Netzwerkkarte im Netz praktisch unsichtbar bleibt.

Man kann auch unter Benutzung eines Programms, das gezielt die Link-Layer-Schnittstellen des Systems nutzt (und dazu root-Rechte benötigt), Pakete ins Netz senden, beispielsweise harmlose ARP-Anfragen, um das Vorhandensein anderer Stationen zu testen. Ein solches Programm ist arping, das von den Distributionen meist im gleichnamigen Paket angeboten wird und die Benutzung unkonfigurierter Schnittstellen unterstützt. Hier ein Beispiel, in dem mit arping über eine WLAN-Karte der Access-Point mit der Adresse 192.168.1.1 angesprochen wird:

```
hugo:~# ip addr show dev ath0
3: ath0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 200
    link/ether 00:05:4e:4c:2b:3f brd ff:ff:ff:ff:ff:ff
    inet6 fe80::205:4eff:fe4c:2b3f/64 scope link
       valid_lft forever preferred_lft forever
hugo:~# arping -Ov -i ath0 -c 3 192.168.1.1
This box:   Interface: ath0  IP: 255.255.255.255   MAC address: 00:05:4e:4c:2b:3f
ARPING 192.168.1.1
42 bytes from 00:0f:66:d9:99:4f (192.168.1.1): index=0 time=3.359 msec
42 bytes from 00:0f:66:d9:99:4f (192.168.1.1): index=1 time=785.112 usec
42 bytes from 00:0f:66:d9:99:4f (192.168.1.1): index=2 time=792.980 usec

--- 192.168.1.1 statistics ---
3 packets transmitted, 3 packets received,   0% unanswered
hugo:~#
```

Auf diese Weise lässt sich feststellen, in welcher Netzwerkumgebung man sich befindet, ohne dass man seinem Rechner eine IP-Adresse gibt, die womöglich schon anderweitig vergeben ist und dadurch den Betrieb stört. Natürlich eignen sich Programme, die im Subnetz kommunizieren, ohne eine eigene IP-Adresse zu verwenden, auch als Hackertools, aber verraten Sie das bitte niemandem … ;-)

A.1.6 Adressenzuweisung

Einer aktivierten Netzwerkschnittstelle kann durch `ip addr add` eine IP-Adresse zugewiesen werden. Normalerweise, z. B. beim Hochfahren des Systems, geschieht

dies unmittelbar nach der Aktivierung. Dies hat zwei wesentliche Nebeneffekte: Die Adresse wird nicht nur der Schnittstelle, sondern auch dem Rechner zugewiesen, und es wird automatisch ein Routing-Eintrag für das Subnetz erzeugt, dem die Adresse angehört.

Damit dieser Routing-Eintrag korrekt ist, muss unbedingt die korrekte Präfixlänge (Netzmaske) angegeben werden. Versäumt man dies, so ist das betreffende Subnetz unerreichbar:

```
hugo:~# ip addr add 192.168.77.12 dev eth0
hugo:~# ip route show dev eth0
hugo:~#
```

Um diesen Zustand in Ordnung zu bringen, kann die Adresse wieder entfernt werden (jetzt aber mit Präfixlänge, sonst erscheint – sinnigerweise erst jetzt – eine Warnung) und neu zugewiesen werden:

```
hugo:~# ip addr del 192.168.77.12/32 dev eth0
hugo:~# ip addr add 192.168.77.12/24 dev eth0
hugo:~# ip route show dev eth0
192.168.77.0/24  proto kernel  scope link  src 192.168.77.12
hugo:~#
```

Damit ist jetzt auch eine Route ins Subnetz definiert, und dessen andere Stationen können angesprochen werden.

Dass die vergebene IP-Adresse auch dem Rechner selbst zugeordnet wird, ist daran zu erkennen, dass das Routing für diese Adresse nun auf die rechnerinterne Loopback-Schnittstelle zeigt:

```
hugo:~# ip route get 192.168.77.12
local 192.168.77.12 dev lo  src 192.168.77.12
    cache   mtu 16436 advmss 16396 hoplimit 64
hugo:~#
```

Dies gilt unabhängig von der gerade konfigurierten Schnittstelle: Jedes Paket mit der Zieladresse 192.168.77.12 wird lokal zugestellt (sofern es möglich ist, also ein Programm bereit ist, die Daten anzunehmen).

A.1.7 Lokale Adressen als Ziel

Grundsätzlich werden Pakete an eine lokale Adresse von jeder Schnittstelle aus angenommen. Deshalb kann man auch Pakete an jede solche Adresse von einem Programm, das auf demselben Rechner läuft, verschicken, z. B. werden

```
ping localhost
ping 127.0.0.1
```

ebenso an das Loopback-Interface `lo` geschickt (und im Falle des Ping schon vom Linux-Netzwerkcode beantwortet) wie

```
ping 192.168.42.7
```

wenn dies eine Adresse ist, die einer (echten oder simulierten, also auch Dummy-) Netzwerkkarte zugewiesen wurde.

Deshalb würde auch ein Paket angenommen, das über eine externe Schnittstelle ankommt und die Zieladresse 127.0.0.1 hat. Aber in diesen schlechten Zeiten, da das Internet voller Bösewichter steckt, könnte mit solchen Paketen Schindluder getrieben werden: Dem Empfangsrechner würde vorgegaukelt, es handele sich um ein lokales Paket, und er würde es vielleicht leichtfertig akzeptieren.

Darum schaltet man möglichst immer den Reverse-Path-Filter ein. Dabei handelt es sich um eine Beschränkung für eine Netzwerkschnittstelle, die alle Pakete abweist, deren Absenderadresse nicht auch bei umgekehrt laufendem Datenverkehr als Ziel über dieselbe Schnittstelle geroutet würde. Auf gut Deutsch heißt das, Pakete werden nur von dort angenommen, wo die Antwort auch hingeschickt würde. Alles andere ist – im Normalfall – Teufelszeug und soll falsches Vertrauen erwecken. Wenn allerdings die Netzwerktopologie so kompliziert ist, dass die Pakete sozusagen auf Schleichwegen ankommen und in Hin- und Rückrichtung tatsächlich unterschiedliche Wege nehmen, dann muss dieser Filter abgeschaltet werden.

Im Zweifelsfall aktivieren Sie global den Reverse-Path-Filter durch

```
echo 1 > /proc/sys/net/ipv4/conf/default/rp_filter
```

oder durch den Eintrag

```
net.ipv4.conf.default.rp_filter=1
```

in der Datei `/etc/sysctl.conf`. Sie können den Filter auch pro Interface setzen. Bei Debian-Systemen können Sie ihn beispielsweise durch eine Zeile `ip-rp-filter 0` in der zugehörigen Strophe der Netzwerk-Konfigurationsdatei `/etc/network/interfaces` gezielt abschalten.

A.1.8 Mehrere Adressen für ein Interface

Alte Unix-Hasen sehen zwischen Netzwerkschnittstellen und IP-Adressen oft noch einen 1:1-Zusammenhang und wollen eine zweite Netzwerkkarte einbauen, wenn ihr Rechner zwei unterschiedliche Adressen in demselben physikalischen Netzwerksegment bekommen soll. Für Linux-Systeme ist das längst nicht mehr nötig, man muss nicht einmal einen zweiten (Alias-)Namen für das Interface erfinden: Die Kommandosyntax `ip add ...` weist schon darauf hin, dass eine Adresse dem Interface *hinzugefügt* werden soll. Meist handelt es sich zwar um die erste und einzige Adresse, aber es spricht nichts dagegen, dem Interface zwei oder auch zwanzig Adressen zu spendieren.

Dies ist nicht nur eine hübsche Spielerei, sondern hat durchaus einen Sinn: Sie können für unterschiedliche Services, die auf demselben Rechner laufen, verschiedene IP-Adressen definieren und die Programme entsprechend konfigurieren, sodass beispielsweise der Webserver auf einer und der Mailserver auf einer anderen Adresse lauscht. Bei Bedarf können Sie später diese Anwendungen auf zwei Rechner verteilen, von denen jeder eine der Adressen übernimmt. Damit bleibt die Änderung ohne Auswirkungen auf die Clients, die den Web- und den Mailserver nach wie vor unter der gewohnten Adresse ansprechen.

Vergabe von Absenderadressen

Bei Benutzung mehrerer Adressen auf einem Interface muss Linux freilich wissen, welche Absenderadresse es in die Pakete eintragen soll, die das System über dieses Interface verlassen. Am sichersten ist es, wenn das Anwendungsprogramm, das die Daten losschicken soll, beim Binden an das Interface eine bestimmte Adresse festlegt. Viele Netzwerkprogramme sehen dies in ihrer Konfiguration vor (beispielsweise auch der Apache-Webserver oder der ssh-Dämon).

Sonst sucht sich das System eine Adresse aus, die möglichst zum selben Subnetz gehört wie die Zieladresse oder der nächste Router; dies kann im Zweifelsfall sogar eine Adresse sein, die gar nicht diesem Interface zugeordnet ist. Machen Sie deshalb von der Möglichkeit mehrfacher Adressen im Regelfall nur Gebrauch, wenn alle diese Adressen in dasselbe Subnetz fallen, und konfigurieren Sie die Serverprogramme auf jeweils eine der Adressen.

A.2 Netzwerkprotokolle

Sie haben sich nun bis zum Schluss dieses Buchs durchgearbeitet (oder -geblättert?) und vermissen noch dieses oder jenes Netzwerkprotokoll, das wir nicht behandelt haben? Vielleicht haben wir es übersehen, vielleicht aber auch bewusst ausgelassen, weil wir es für nicht mehr zeitgemäß oder zu unsicher halten, um es noch ohne Not anzuwenden. Einige durchaus nützliche Protokolle konnten wir aus Platzgründen nicht ausführlich beschreiben.

Nachfolgend haben wir kurze Steckbriefe von bekannten, in diesem Buch aber nicht näher behandelten Protokollen zusammengestellt, um Ihnen darzulegen, wozu sie überhaupt benutzt werden. Außerdem haben wir die Gelegenheit genutzt, unsere subjektive Meinung darüber loszuwerden, ob diese Protokolle überhaupt noch eine Daseinsberechtigung haben und welche von ihnen man nach Möglichkeit durch Firewall-Regeln abblocken sollte.

echo

Standardport: 7 (TCP, UDP)

Bedeutung: Der `echo`-Service verhält sich sehr ähnlich zu dem, der den Touristen am Königssee geboten wird: Er sendet die Daten, die er

empfängt, postwendend (als Echo) wieder zurück. Dies kann zu Testzwecken ganz nützlich sein.

Implementation: Oft als eingebaute Funktion des `inetd` (siehe Abschnitt 11.3.3) realisiert.

Sicherheit: Dieser Dienst lässt sich leicht missbrauchen, um die Funktion eines Netzes zu stören.

Anmerkung: `echo` sollte in der `inetd`-Konfiguration nicht aktiviert sein.

Das Gleiche gilt für eine Reihe ähnlicher Dienste: `discard`: Port 9, `quote`: Port 17, `chargen`: Port 19.

daytime

Standardport: 13 (TCP, UDP)

Bedeutung: Über `daytime` kann die Uhrzeit (der rechnerinternen Uhr) als String abgefragt werden.

Implementation: Oft als eingebaute Funktion des `inetd` (siehe Abschnitt 11.3.3) realisiert.

Sicherheit: Dieser Dienst lässt sich leicht missbrauchen, um die Funktion eines Netzes zu stören.

Anmerkung: `daytime` sollte in der `inetd`-Konfiguration nicht aktiviert sein. Mit `ntp` gibt es eine sehr viel bessere (und genauere) Alternative.

Das Gleiche gilt für `time`: Port 37 (Abfrage der Uhrzeit in binärem Format).

telnet

Standardport: 23 (TCP)

Bedeutung: Mit `telnet` kann man sich auf einen entfernten Rechner einloggen und einen Shell-Prompt fernbedienen.

Implementation: Der Telnet-Dämon `telnetd` wird meist über den Internet-Dämon gestartet (siehe Abschnitt 11.3.3).

Sicherheit: Das Telnet-Protokoll ist hochgradig unsicher; Passwörter werden im Klartext übertragen (und zur Unterstützung böser Hacker durch eine vorangestellte Zeichenfolge `PASS` besonders markiert). Es gibt durch SSL gesicherte Varianten dieses Protokolls, die aber keine große Verbreitung gefunden haben.

Anmerkung: Der Telnet-Dämon sollte nicht installiert sein.

Das Client-Programm `telnet` kann dennoch gute Dienste leisten, weil man mit ihm zu Testzwecken Verbindungen zu beliebigen TCP-Ports herstellen und textorientierte Protokolle wie HTTP und SMTP »zu Fuß« benutzen kann.

whois

Standardport: 43 (TCP)

Bedeutung: Mit `whois` können die Datenbanken abgefragt werden, in denen die Inhaber von Domain-Namen und IP-Adressbereichen registriert sind.

Implementation: Normalerweise wird nur das Clientprogramm `whois` installiert.

Sicherheit: Keine besondere Gefährdung.

Anmerkung: `whois` kann bei Bedarf ohne Bedenken in Richtung des Internet durch Firewall-Regeln zugelassen werden. Ankommende Verbindungen sind nicht sinnvoll und sollten abgewiesen werden.

gopher

Standardport: 70 (TCP, UDP)

Bedeutung: `gopher` war ein früher Vorläufer des World Wide Web.

Implementation: Die spezifischen Client- und Serverprogramme sind obsolet. Einige Browser und Programmiersprachen bieten noch Unterstützung für dieses Protokoll.

Sicherheit: Das Protokoll bietet keine gesicherte Übertragung.

Anmerkung: Der `gopher`-Port braucht bei einer Firewall normalerweise nicht freigegeben zu werden.

finger

Standardport: 79 (TCP)

Bedeutung: Das `finger`-Programm diente ursprünglich als billiger Ersatz für einen Verzeichnisdienst: Es informierte über wichtige Kenndaten einer Benutzerkennung und konnte hierzu auch fremde Rechner befragen. Auch konnte man erfahren, welche Benutzer gerade eingeloggt waren. Es lässt sich übrigens auch auf dem lokalen Rechner anwenden: Probieren Sie das Kommando `finger` ohne Parameter oder mit Angabe einer Kennung aus!

Implementation: Es gibt `fingerd` und mehrere weitere, etwas sicherere Implementationen eines `finger`-Dämons.

Sicherheit: Die von `finger` bereitgestellte Information kann für Angreifer ausgesprochen nützlich sein. Außerdem bietet das Protokoll selbst keine Sicherheit.

Anmerkung: Der `finger`-Dämon sollte möglichst gar nicht installiert werden; der verwendete Port sollte durch Firewall-Regeln blockiert sein.

kerberos

Standardport: 88 (TCP, UDP)

Bedeutung: Kerberos ist ein zentraler Autorisierungsdienst, der von verschiedenen anderen Diensten genutzt werden kann.

Implementation: Es gibt eine Reihe verschiedener Programme und Bibliotheken, die Kerberos implementieren.

Sicherheit: Kerberos gilt als sehr sicheres Protokoll.

Anmerkung: Der zentrale Kerberos-Server sollte durch Firewalling gegen Angriffe über andere Dienste bzw. Protokolle gut geschützt werden.

Portmapper

Standardport: 111 (UDP, TCP)

Bedeutung: Der Portmapper stellt eine einheitliche Schnittstelle dar, um andere Dienste zu vermitteln, die über dynamisch zugewiesene Ports abgewickelt werden.

Implementation: Dieser Dienst wird durch einen Dämon realisiert (oft `/sbin/portmap`), der auf jedem Rechner laufen muss, der über den Portmapper vermittelte Dienste anbietet.

Sicherheit: Das Portmapper-Protokoll ist nicht besonders geschützt; die Sicherheit ist aber vorrangig von den jeweiligen Diensten zu gewährleisten.

Anmerkung: Das Portmapper-Konzept hat nicht die universelle Verbreitung gefunden, die sich seine Entwickler vermutlich erhofft hatten. Zum Betrieb eines NFS-Servers ist der Portmapper erforderlich, weil die Clients zunächst über Port 111 zugreifen und sich nach dem NFS-Port erkundigen; dieser ist allerdings in der Regel festgelegt (auf 2049). Damit »portgemappte« Dienste durch eine Firewall hindurch benutzbar sind, müssen alle zugehörigen Ports festgelegt werden (unter Linux ist das meist möglich), und die Firewall-Regeln können dann den Zugriff über diese Ports erlauben.

NNTP

Standardport: 119 (TCP)

Bedeutung: NNTP (Network News Transfer Protocol) wird zum Austausch von Nachrichten für Diskussionsforen (sogenannte Usenet-Gruppen) verwendet. Zum Lesen der Beiträge und zum Versand eigener Weisheiten können spezielle Clientprogramme (»Newsreader«) und auch viele Webbrowser benutzt werden. Die Nachrichten sind – im Gegensatz zum WWW – überwiegend textorientiert.

Implementation: Die Serverprogramme (z. B. `nntpd`) erhalten die Nachrichten aus dem Usenet meist von zentralen NNTP-Servern, die von Providern betrieben werden und bei denen sie als »Feed« angemeldet werden müssen, und sie liefern die von lokalen Benutzern geschriebenen Beiträge dort ab.

Sicherheit: Das NNTP-Protokoll bietet keine gesicherte Übertragung.

Anmerkung: Das Usenet hat durch das Entstehen von Diskussionsforen im Web viel an Bedeutung verloren, wird aber von einigen »hartgesottenen« Anwendern noch intensiv genutzt. Soll ein NNTP-Server alle relevanten Newsgruppen führen, so wird ein recht hohes Datenaufkommen entstehen.

Im Normalfall sollte die Nutzung eines NNTP-Servers auf lokale Rechner beschränkt werden, weil der Server sonst eine unüberschaubare Zahl von Nutzern anziehen könnte. Dies mag aus Kapazitätsgründen und im Hinblick auf die Verantwortung für eingelieferte Beiträge nicht immer erwünscht sein.

SNMP

Standardport: 161, 162 (TCP, UDP)

Bedeutung: SNMP, das Simple Network Management Protocol, dient zur Fernwartung von Netzwerkkomponenten, deren Zustandsdaten darüber abgefragt werden können und die über sogenannte SNMP-Traps auch asynchrone Ereignisse melden.

Implementation: Es sind zahlreiche SNMP-Client- und Serverprogramme sowie Bibliotheken verfügbar. Viele Geräte wie z. B. Drucker, Router oder Switches bieten SNMP-Funktionalität an.

Sicherheit: SNMP verfügt – zumindest in den meist implementierten und verwendeten älteren Versionen – nicht über wirksame Sicherheitsfunktionen.

Anmerkung: Gemäß seiner Zweckbestimmung sollte SNMP nur im lokalen Netz benutzt werden; durch Firewalling kann daher der Zugriff auf die SNMP-Ports eng eingeschränkt werden. Sofern die Implementationen über Sicherheitsfeatures verfügen, sollten sie diese nutzen. Zumindest sind die als einfache Passwörter fungierenden »Community Strings« fantasievoll abzuändern.

LDAP

Standardport: 389, 636 (TCP)

Bedeutung: LDAP ist das Lightweight Directory Access Protocol. Es ermöglicht den Zugriff auf Verzeichnisserver, die beispielsweise zur Verifikation von Benutzer-Logins genutzt werden können.

Implementation: Unter Linux wird häufig OpenLDAP als LDAP-Server eingesetzt. Es sind zahlreiche Client-Programme und -Bibliotheken

verfügbar. Die meisten Linux-Systeme können über PAM (»Pluggable Authentication Modules«) für die Benutzung von LDAP (z. B. zur Benutzerauthentifizierung) konfiguriert werden.

Sicherheit: Ältere LDAP-Versionen müssen als unsicher gelten und sollten (wegen der Brisanz der übertragenen Daten) über SSL-Verbindungen oder VPN getunnelt werden. Die LDAP-Version 3 bietet verbesserte Sicherheitsmechanismen.

Anmerkung: Um beispielsweise externen Benutzern das Ausprobieren von Passwörtern zu verwehren, sollte der Zugriff auf LDAP-Server durch Firewall-Regeln beschränkt werden.

r-Services

Standardport: 512, 513, 514 (TCP)

Bedeutung: Die Protokolle `rexec`, `rlogin` und `rshell` stellen Funktionen bereit, die inzwischen auch von der

Implementation: Die serverseitigen Programme werden in der Regel über den Internet-Dämon gestartet.

Sicherheit: Diese Dienste sind hochgradig unsicher und sollten nicht mehr verwendet werden. Funktional können sie leicht durch die Secure Shell abgelöst werden.

Anmerkung: Die Dämonen sollten nicht gestartet werden (und nicht in der `inetd`-Konfiguration aktiviert sein). Die von der `ssh` gebotenen Kompatibilitätsfunktionen sollten in den Konfigurationsdateien abgeschaltet bleiben. Die zugehörigen Portnummern können in Firewalls blockiert werden, soweit sie nicht von anderen Protokollen (z. B. `syslog`) verwendet werden.

Radius

Standardport: 812 (TCP, UDP)

Bedeutung: Das Radius-Protokoll dient, ähnlich wie das (modernere) LDAP, der Benutzerauthentifizierung, außerdem zur Übermittlung von Accounting-Daten.

Implementation: Für Radius-Server existiert mit FreeRadius eine Open-Source-Implementation.

Sicherheit: Das Radius-Protokoll bietet wenig Sicherheit; es sollte gegebenenfalls über SSL oder VPN getunnelt werden.

Anmerkung: Der Zugriff auf Radius-Server sollte durch Firewalling auf berechtigte Clients begrenzt werden.

rsync

Standardport: 873 (TCP, UDP)

Bedeutung: Mit `rsync` können Datei- und Verzeichnisinhalte zwischen zwei Rechnern abgeglichen werden. Das Protokoll überträgt nach Möglichkeit nur veränderte Inhalte und ist dadurch recht effizient.

Implementation: Das Programm `rsync` übt sowohl die Client- als auch die Serverfunktion aus. Letztere kann es als Dämon oder auch als Serverprozess ausüben, der von `inetd` gestartet wird.

Sicherheit: `rsync` bietet eine Vielzahl von Optionen, durch die es für unterschiedlich sichere Betriebsarten konfiguriert werden kann.

Anmerkung: Nach Möglichkeit sollte `rsync` so aufgerufen werden, dass es die Secure Shell zur Übertragung verwendet. Die Manpage gibt hierfür einige Beispiele.

Stichwortverzeichnis

InformIT™
CHECK IT

... aktuelles Fachwissen rund
um die Uhr – zum Probelesen,
Downloaden oder auch auf Papier.

www.InformIT.de

InformIT.de, Partner von **Addison-Wesley**, ist unsere Antwort auf alle Fragen der IT-Branche.

In Zusammenarbeit mit den Top-Autoren von Addison-Wesley, absoluten Spezialisten ihres Fachgebiets, bieten wir Ihnen ständig hochinteressante, brandaktuelle Informationen und kompetente Lösungen zu nahezu allen IT-Themen.

wenn Sie mehr wissen wollen ... **www.InformIT.de**

Copyright

Daten, Texte, Design und Grafiken dieses eBooks, sowie die eventuell angebotenen eBook-Zusatzdaten sind urheberrechtlich geschützt. Dieses eBook stellen wir lediglich als persönliche Einzelplatz-Lizenz zur Verfügung!

Jede andere Verwendung dieses eBooks oder zugehöriger Materialien und Informationen, einschliesslich

- der Reproduktion,
- der Weitergabe,
- des Weitervertriebs,
- der Platzierung im Internet, in Intranets, in Extranets,
- der Veränderung,
- des Weiterverkaufs
- und der Veröffentlichung

bedarf der schriftlichen Genehmigung des Verlags.

Insbesondere ist die Entfernung oder Änderung des vom Verlag vergebenen Passwortschutzes ausdrücklich untersagt!

Bei Fragen zu diesem Thema wenden Sie sich bitte an: info@pearson.de

Zusatzdaten

Möglicherweise liegt dem gedruckten Buch eine CD-ROM mit Zusatzdaten bei. Die Zurverfügungstellung dieser Daten auf unseren Websites ist eine freiwillige Leistung des Verlags. Der Rechtsweg ist ausgeschlossen.

Hinweis

Dieses und viele weitere eBooks können Sie rund um die Uhr und legal auf unserer Website

http://www.informit.de

herunterladen