



KAPITEL 6

Eine Firewall auf einer Floppy

Unser Heimanwender Heinz (siehe Kapitel 2, Unterabschnitt *Der Privathaushalt*, ab Seite 9) hat eher bescheidene Ansprüche an eine Firewall. Weder will er einen eigenen Webserver betreiben, noch will er mit einem Proxy Webseiten zwischenspeichern oder Inhalte filtern. Auch die Anzahl der Nutzer seiner Firewall ist gering. Die einzige Anforderung an seine Firewall besteht darin, daß eventuelle Angreifer aus dem Internet auf die Rechner in seinem lokalen Netz nicht direkt zugreifen können.

In so einer Situation ist es sicherlich eine Überlegung wert, eine Distribution zu verwenden, die direkt von einer Diskette gestartet werden kann und die speziell für den Einsatz als Firewall entwickelt wurde. Sie finden solche Mini-Linuxe auf den bekannten Software-Sites wie

<http://freshmeat.net> (suchen Sie in der Kategorie Topic::System::Operating System nach »router« oder »firewall«)

<http://www.ibiblio.org/pub/Linux/distributions/> (eine Auflistung diverser Distributionen, darunter auch Disketten-Linuxe)

<http://sourceforge.net> (suchen Sie nach »linux router« oder »firewall«)

Sie werden allerdings feststellen, daß nicht alle Vertreter der Gattung gleich gut geeignet für Ihre Zwecke sind. Es ist daher wichtig, genau hinzusehen, was ein Floppy-Linux leistet und was nicht. Für eine erste Vorauswahl reichen oft einige wenige Kriterien:

- Das Linux sollte speziell dafür konzipiert sein, für mehrere Rechner eine Internetverbindung herzustellen. Distributionen, die für Tests der Netzwerksicherheit oder als Surfsysteme gedacht sind, könnten zwar unter Umständen umfunktioniert werden, dies würde aber einen großen Aufwand bedeuten.
- Es sollte Firewalling unterstützen.
- Es sollte die von Ihnen genutzte Einwahlmethode ins Internet unterstützen (z. B. Modem oder ISDN). Es gibt Versionen, die z. B. nur eine direkte Netzwerkanbindung mittels Ethernet unterstützen. Das Kompilieren und Einbinden der fehlenden Software wäre aber wahrscheinlich den Aufwand nicht wert.
- Die von Ihnen verwendete Netzwerkkarte sollte unterstützt werden.

- Das Linux sollte schon eine Weile existieren, und aus den Webseiten des Projekts sollte hervorgehen, daß in letzter Zeit Änderungen daran vorgenommen wurden. Es existieren viele Projekte, die mit viel Eifer begonnen, dann aber nicht weiterentwickelt wurden.

Im folgenden werde ich zwei Linuxe beschreiben, die diese Kriterien erfüllen. Beim ersten handelt es sich um Coyote Linux. Es unterstützt den Zugang über Modem oder DSL. Als minimale Hardware-Anforderung spezifiziert der Autor einen 386SX-Prozessor, 12 MB RAM, ein 3,5-Zoll-HD-Floppylaufwerk und eine MDA-Graphikkarte.

Das zweite Linux ist fli4l. Es bietet keine Unterstützung für Modems und erwartet mindestens einen 386er Prozessor mit 25 MHz Systemtakt. Empfohlen wird für den ISDN-Betrieb aber eher ein 486er mit 33 MHz. Für DSL sollte es dagegen schon ein 486DX2/66, 486DX6/100 oder ein Pentium 75 MHz sein. Grundsätzlich ist der Betrieb mit 12 MB RAM möglich, 16 MB werden empfohlen.

Beide Distributionen benötigen keine Festplatte, sondern starten direkt von einer Diskette. Abgesehen von den unterstützten Zugriffsmethoden liegt der Hauptunterschied der beiden Distributionen in ihrer Philosophie. fli4l richtet sich an erfahrene Linuxbenutzer. In der Grundversion ist es recht spartanisch, aber eine Vielzahl von Zusatzpaketen erlaubt es, das System in alle Richtungen zu erweitern (Web-, FTP-, SSH-Zugriff, Printserver, Anschluß einer externen LCD-Anzeige, Ausführung von Kommandos bei Anruf, ...). Darüber hinaus hat es standardmäßig einen Dienst, der es erlaubt festzulegen, zu welcher Uhrzeit welcher Provider verwendet wird. Das System erinnert an einen Werkzeugkasten, mit dem ein Profi alles bauen kann, was er für den Zugriff auf das Internet benötigt.

Coyote Linux ist dagegen monolithischer. Es glänzt nicht so sehr durch seine Vielseitigkeit als durch seine Bedienungsfreundlichkeit. Der Schwerpunkt liegt hier deutlich darauf, einen möglichst reibungslosen Zugang zum Internet zu gewährleisten. Dies zeigt sich auch darin, daß standardmäßig alle Masquerading-Module geladen werden, die der verwendete Kernel bietet. Auch werden standardmäßig ein Web- und ein SSH-Server installiert. Letzterer kann sogar dazu genutzt werden, die Firewall über das Internet zu administrieren¹. Das Webinterface ist nur aus dem lokalen Netz zugreifbar, es erlaubt es aber auf überaus komfortable Weise, alle Aspekte der Firewall zu steuern.

Beide Distributionen erlauben im Prinzip eine Installation unter Windows. Allerdings ist der Funktionsumfang in diesem Fall unter Umständen etwas eingeschränkt. Ich beschränke mich hier aber auf die Installation unter Linux.

Vorüberlegungen

Auf einer Diskette ist nur begrenzt Platz. Aus diesem Grund besitzen beide Distributionen ein Installationsprogramm, das eine Diskette erzeugt, die nur die Komponenten enthält, die Sie auch tatsächlich benötigen. Sie müssen sich daher vor der Installation

¹ Keine Sorge, wir werden sehen, wie man das abstellt.

die nötigen Informationen über die technischen Daten Ihres Zielsystems, Ihres lokalen Netzes und Ihrer Netzanbindung zusammenstellen.

Hier ein kleiner Überblick:

Prozessor Hier ist insbesondere wichtig, ob ein mathematischer Koprozessor fehlt. D. h., handelt es sich um einen 386SX, 386 oder 486SX?

Netzwerkkarte(n) Wie heißt das zuständige Modul? Ein erster Ansatz kann hier die Netzwerkkartendatenbank von fli4l sein:

<http://www.fli4l.de/german/extern/nics/index.php>

ISDN-Karte Die Daten zur Konfiguration einer ISDN-Karte sind oft nicht ganz intuitiv. Hier hilft ein Blick in die Dokumentation zum HiSax-Treiber. Sie finden sie in den Kernelquellen unter

`usr/src/linux/Documentation/isdn/README.HiSax`

Wenn Sie die Kernelquellen nicht installiert haben, so finden Sie das Dokumentationsverzeichnis auch unter

<http://www.linuxhq.com>

Im einzelnen benötigen Sie die folgenden Angaben:

HiSax-Nummer Diese Nummer teilt dem Treiber mit, was für eine ISDN-Karte Sie benutzen. Eine Fritz!Card ISA (nicht PnP) hat z. B. die Nummer 5, während PnP- und PCI-Karten desselben Herstellers die Nummer 27 haben.

Die Fritz!Card PCI v2 wird von HiSax nicht unterstützt. Wenn Sie aber statt der hier beschriebenen stabilen fli4l-Version 2.0.8 eine aktuelle Betaversion (2.1.x) verwenden, so können Sie sie unter der Nummer 39 benutzen.

IRQ bei PCI-Karten oft unnötig.

Port bei PCI-Karten oft unnötig.

Speicheradresse bei PCI-Karten oft unnötig.

Schicht-2-Protokoll Zur Auswahl stehen (Protokollnummer in Klammern):

- hdlc (1)
- x75i (2)
- x75ui (3)
- x75bui (4)

Welches Protokoll verwendet wird, erfahren Sie von Ihrem Provider, allerdings stehen die Chancen gut, daß es sich um hdlc handelt.

MSN In Deutschland ist dies in der Regel Ihre Telefonnummer ohne Vorwahl.

Lokales Netz Ihre Firewall muß wissen, welche Adressen die Rechner in Ihrem lokalen Netz benutzen. Dabei sollte es sich möglichst um Adressen aus einem Bereich handeln, der für private Subnetze reserviert ist (siehe Kapitel 3, Abschnitt *IP*, ab Seite 20). Konkret benötigen Sie die folgenden Angaben:

- Adresse der Netzwerkkarte (z. B. 192.168.20.1)
- Netzmaske (z. B. 255.255.255.0)
- Broadcast-Adresse (z. B. 192.168.20.255)
- »Netzwerk-Adresse« (z. B. 192.168.20.0)

Angaben zur Internet-Anbindung Diese Daten erhalten Sie normalerweise von Ihrem Provider.

Externe IP-Adresse Hat der Provider Ihnen eine statische IP-Adresse eingerichtet, so nehmen Sie diese. Ansonsten wählen Sie eine Adresse aus einem der Adreßbereiche, die für die Nutzung durch private Subnetze vorgesehen sind (z. B. 10.1.0.1). Achten Sie aber darauf, daß die Adresse nicht aus dem Bereich stammt, den Sie für Ihr lokales Netz benutzen.

Telefonnummer Die Nummer Ihres Providers.

DNS-Server Normalerweise stellt Ihnen der Provider ein oder zwei Name-Server zur Verfügung.

Domäne Die DNS-Domäne, in der sich Ihre Rechner befinden. Wenn Sie nicht wissen, was Sie hier einstellen sollen, können auch fiktive Werte wie »nowhere.invalid« angegeben werden, ohne daß der Zugang darunter leidet.

Username Ihre Benutzerkennung.

Paßwort Ihr Paßwort.

Für erste Gehversuche sind in Anhang A ab Seite 595 einige Provider aufgeführt, die einen Zugang ohne vorherige Registrierung erlauben. Die Bezahlung erfolgt einfach über die Telefonrechnung.

Coyote Linux

Kommen wir nun zur Installation von Coyote Linux. Diese besteht im wesentlichen darin, eine Diskette einzulegen, ein paar Fragen zu beantworten und dann die eigentliche Firewall von dieser Diskette zu booten. Allerdings hat Coyote Linux bei näherem Hinsehen ein paar kleinere Schönheitsfehler. Aus diesem Grund werden wir später in diesem Kapitel auf diese Probleme eingehen und sehen, wie sie gelöst werden können.

Installation

Coyote Linux können Sie von <http://www.coyotelinux.com> herunterladen. Es handelt sich dabei um eine Datei (hier: *coyote-2.13.tar.gz*).

Für die nun folgende Installation benötigen Sie Rootrechte. Packen Sie als erstes das Archiv aus:

```
# tar xvzf coyote-2.13.tar.gz
```

Damit existiert nun ein neues Verzeichnis *coyote*, in dem Sie das Shellskript *makefloppy.sh* finden. Sie müssen es nur ausführen, und es wird ein auf Ihre Bedürfnisse angepaßtes System generiert:

```
# cd coyote
# ./makefloppy.sh
Coyote floppy builder script v2.9
```

Nun folgt eine Reihe von Fragen, die bestimmen, was letztlich auf die Diskette kopiert wird. Als erstes bietet Ihnen das Installationsprogramm an, eine Diskette so zu formatieren, daß mehr als die üblichen 1,44 MB darauf Platz finden. Dieses Angebot sollten Sie annehmen und mit mindestens 1,68 MB formatieren:

```
Please choose the desired capacity for the created floppy:

1) 1.44Mb (Safest and most reliable but may lack space needed for
   some options)
2) 1.68Mb (Good reliability with extra space) - recommended
3) 1.72Mb (Most space but may not work on all systems or with all
   diskettes)

Enter selection: 2
```

Nun gilt es zu entscheiden, wie das System ans Internet angebunden werden soll. Derzeit bietet Coyote die direkte Netzanbindung (Standard Ethernet Connection), DSL (PPP over Ethernet Connection) und die Anbindung über Modem (PPP Dialup Connection).

Im folgenden soll die Anbindung über Modem beschrieben werden. Die Konfiguration der anderen Anbindungen unterscheidet sich aber kaum von der hier beschriebenen, es werden lediglich zusätzliche Daten für die Netzwerkkarte abgefragt, über welche die Anbindung an das Internet erfolgt:

```
Please select the type of Internet connection that your system uses.

1) Standard Ethernet Connection
2) PPP over Ethernet Connection
3) PPP Dialup Connection
4) ISDN Connection (experimental)

Enter Selection: 3

Configuring system for PPP dialup.
```

Nun gilt es festzulegen, welche Adressen Ihr lokales Netz benutzt. Vorgabe ist hier *192.168.0.x*, man kann aber auch andere Netze einstellen (hier: *192.168.20.x*):

```
By default, Coyote uses the following settings for the local network interface:

IP Address: 192.168.0.1
Netmask:    255.255.255.0
Broadcast:  192.168.0.255
Network:    192.168.0.0
```

```
Would you like to change these settings? [Y/N]: y
Enter local IP Address [192.168.0.1]: 192.168.20.1
Enter local Netmask [255.255.255.0]: 255.255.255.0
Enter local Broadcast [192.168.0.255]: 192.168.20.255
Enter local network number [192.168.0.0]: 192.168.20.0
```

Wenn Sie nicht gerade eine Flatrate nutzen, werden Sie sicherlich wollen, daß die Firewall nur bei Bedarf die Verbindung zum Internet herstellt und auch wieder auflegt, wenn eine Weile keine Daten übertragen werden:

```
OPTIONS CONFIGURATION
Demand Dial:
Initiate the link only on demand, i.e. when data
traffic is present.
```

```
Do you want to enable the demand dial option [y/n]: y
```

Wie lange die Firewall auf Anfragen wartet, bis sie die Verbindung wieder abbaut, ist Geschmackssache. Allerdings würde ich die Zeit bei einer Modemanbindung nicht zu kurz wählen, da Wahlvorgänge deutlich länger dauern als bei ISDN. Außerdem existieren Provider, die die erste Anfrage nach einem Verbindungsaufbau auf eine eigene Seite umleiten. Wenn man die Zeit also zu kurz wählt, dann kann es einem passieren, daß man nach jedem Lesen einer Seite erst einmal mehrere Versuche braucht, um auf die nächste Seite zu kommen, nur um dann festzustellen, daß man anstatt am gewünschten Ziel wieder auf der Homepage seines Providers gelandet ist.

Hier habe ich einmal eine Wartezeit von zehn Minuten vorgegeben:

```
Idle option:
Specifies that pppd should disconnect if the link
is idle for n seconds. The link is idle when no
data packets (i.e. IP packets) are being sent or
received.
```

```
Do you want to enable the idle option [y/n]: y
```

```
Enter number of seconds for idle disconnect [180]: 600
```

Wahrscheinlich wird Ihr Provider Ihnen dynamische Adressen zuweisen. Sie können daher die nächste Frage verneinen:

```
Did your ISP assign you a static IP ADDRESS? [y/n]: n
Setting up for dynamic PPP Address
```

Zwar weist der Provider Ihnen nach dem Verbindungsaufbau eine IP-Adresse zu, Sie benötigen aber schon vorher eine, um bei Bedarf eine Einwahl realisieren zu können. Sie müssen daher eine Platzhalter-Adresse definieren, die nach der Einwahl durch die vom Provider zugewiesene Adresse ersetzt wird. Es sollte sich dabei um eine Adresse aus einem Bereich für die private Nutzung handeln. Sie sollte aber im Gegensatz zum Vorschlag **nicht** aus dem Subnetz stammen, das Sie in Ihrem lokalen Netz verwenden:

Set the local PPP interface IP address. Should not be the same as 192.168.20.1, but on the same subnet.
Press enter for [192.168.0.3]: 10.1.0.1

Um das Modem nutzen zu können, müssen Sie angeben, an welcher seriellen Schnittstelle es angeschlossen ist (ttyS0 = COM1, ttyS1 = COM2). Die anderen Einstellungen sollten unkritisch sein:

Enter tty device name for modem (ttyS0, ttyS1, etc)[ttyS0]:

Enter ttyS0's port speed (115200, 57600, etc)[115200]:

Enter modem init string (Enter = AT&FS11=55):

Nun werden wir gebeten, die Einwahldaten für unseren Provider anzugeben. Ersetzen Sie die Beispiele bitte durch die Daten des von Ihnen gewählten Providers. Der ISP »dummy-net« ist frei erfunden:

Enter name of ISP (no whitespace)[isp]: dummynet

Enter phone number to dial: 1234567

Enter username: dummy

Enter password: SeCret

Früher war es bei einigen Universitäten notwendig, sich zuerst mit einem Terminalprogramm einzuwählen und mit Name und Paßwort zu authentisieren, bevor dann der pppd gestartet wurde. Normale Provider benutzen diese Methode aber heute nicht mehr:

Does your provider need you to authenticate before running PPP?
ex. if your provider supports neither chap nor pap authentication.

If you enable this, your password will be sent in clear text over the line. Say yes here only if despite having verified everything, you still cannot connect to your ISP.
Login during chat? [y/n]: n

Coyote Linux bietet auch die Möglichkeit, einen DHCP-Server aufzusetzen. Dieser dient dazu, den Rechnern im lokalen Netz automatisch IP-Adressen zuzuweisen. Gerade in großen Netzen ist dies eine Arbeitserleichterung, weil man so einen neuen Rechner einfach an das Netz anschließen kann, ohne darüber nachzudenken, welche IP-Adressen bereits belegt sind. Auch kann man einen Rechner an mehreren Standorten benutzen, obwohl dort unterschiedliche IP-Adressen verwendet werden. Dies ist besonders für Vertreter und Manager vorteilhaft, die von einer Besprechung zur nächsten hetzen. Sie können überall ihren Laptop einstecken und sofort ihre Präsentationen vorführen.

In unserem Szenario gehen wir davon aus, daß wir nur wenige Klientenrechner haben, deren IP-Adressen alle im selben Adreßbereich liegen. Auch wird die Fluktuation gering sein, da man sich normalerweise höchstens alle zwei Jahre einen neuen Rechner kauft. Da ist der zusätzliche Aufwand gering, bei der Installation eine IP-Adresse für den

Rechner anzugeben. Auch hat man bei einer Handvoll von Rechnern kein Problem, den Überblick zu behalten, welche IP-Adressen noch frei sind.

Darüber hinaus ist es generell nicht sinnvoll, auf einem Coyote-System Server zu installieren, da es in der Standardinstallation keinerlei Filterung vornimmt. Jeder Server ist damit auch aus dem Internet zu erreichen. Wir wollen daher auf den DHCP-Server verzichten:

```
Do you want to enable the coyote DHCP server [y/n]: n
```

Mittlerweile bietet Coyote auch die Möglichkeit, eine DMZ aufzusetzen. In unserem Szenario ist dies aber nicht nötig:

```
If you don't know what a DMZ is, just answer NO
You you like to configure a De-Militarized Zone? [Y/N]: n
```

Um logische Adressen wie *www.oreilly.de* in IP-Adressen umsetzen zu können, müssen Sie die zuständigen DNS-Server Ihres Providers angeben. Die hier angeführten Beispiele sind wieder frei erfunden:

```
Enter Domain Name: nowhere.invalid
Enter DNS Server 1: 10.0.0.77
Enter DNS Server 2 (optional): 10.0.0.88
```

Systemmeldungen können auch auf einem anderen Rechner im Netz protokolliert werden. In unserem Fall ist aber unwahrscheinlich, daß im LAN ein extra Syslog-Server betrieben wird. Wir können daher auf diese Möglichkeit verzichten:

```
If you have a syslog server on your LAN you want Coyote to send its syslog
data to, you can specify the address here. If unsure or you do not have a
syslog server, leave this entry blank.
```

```
Syslog server address:
```

Nun braucht das Installationsprogramm nur noch den Namen des Treibers für Ihre Netzwerkkarte sowie bei ISA-Karten unter Umständen den Port und Interrupt. In meinem Beispielrechner steckt eine uralte NE2000-kompatible Karte. Der Treiber für diese ist ne:

```
Enter the module name for your local network card: ne
Enter IO address (Leave blank for PCI cards): 300
Enter IRQ (Leave blank for PCI cards): 9
```

Bei heutigen PCI-Karten entfällt in der Regel die Angabe von IO-Adresse und IRQ.

Nun ist alles konfiguriert, und die Installation kann beginnen. Legen Sie schon einmal eine leere Diskette bereit:

```
Checking module dependencies...
ne deps = 8390
```

```
Building package: etc
Building package: local
Building package: modules
Building package: root
Building package: dhcpd
Building package: webadmin
```

```
Make sure that you have a floppy in the first floppy drive
in this system and press enter to continue...
```

```
Formatting /dev/fd0u1680
Doppelseitig, 80 Spuren, 21 Sektoren/Spur, Totale Kapazität: 1680kB.
Überprüfen ...
Beendet
bin/mkdosfs 2.2 (06 Jul 1999)
Installing boot loader...
Copying files...
cp: Verzeichnis floppy/config\ ausgelassen
floppy/dhcpd.tgz -> mnt/dhcpd.tgz
floppy/etc.tgz -> mnt/etc.tgz
floppy/linux -> mnt/linux
floppy/local.tgz -> mnt/local.tgz
floppy/modules.tgz -> mnt/modules.tgz
floppy/root.tgz -> mnt/root.tgz
floppy/syslinux.cfg -> mnt/syslinux.cfg
floppy/SYSLINUX.DPY -> mnt/SYSLINUX.DPY
floppy/webadmin.tgz -> mnt/webadmin.tgz
floppy/config/coyote.cfg -> mnt/config/coyote.cfg
floppy/config/fireloc.cfg -> mnt/config/fireloc.cfg>
floppy/config/firewall.cfg -> mnt/config/firewall.cfg
floppy/config/hosts.dns -> mnt/config/hosts.dns
floppy/config/portfw.cfg -> mnt/config/portfw.cfg
floppy/config/qosclass.cfg -> mnt/config/qosclass.cfg
floppy/config/qosfilt.cfg -> mnt/config/qosfilt.cfg
floppy/config/reserve.cfg -> mnt/config/reserve.cfg
```

Jetzt enthält die Diskette ein Mini-Linux, das auf die Hardware Ihres Firewallrechners abgestimmt ist. Bei Bedarf können Sie auch noch mehrere Duplikate erzeugen:

```
Would you like to create another copy of this disk [y/n]? n
```

Die so erzeugte Diskette können Sie nun in den eigentlichen Firewallrechner einlegen und den Rechner von ihr booten. Die Anmeldung erfolgt dann als root. Ein Paßwort benötigen Sie dafür noch nicht.

Ein erster Rundgang

Nach der Anmeldung wird ein Menü angezeigt, mit dem Sie diverse Aspekte der Systemkonfiguration einsehen und bei Bedarf ändern können:

Coyote Linux Gateway -- Configuration Menu

- 1) Edit main configuration file
- 2) Change system password
- 3) Edit rc.local script file
- 4) Custom firewall rules file
- 5) Edit firewall configuration
- 6) Edit port forward configuration

- c) Show running configuration
- f) Reload firewall
- r) Reboot system
- w) Write configuration to disk

- q) Exit Menu
- l) Logout

Selection:

Die Punkte 1, 3, 4, 5 und 6 öffnen jeweils ein Untermenü, das Ihnen erlaubt, Konfigurationskripte auszuwählen und in einem Editor zu bearbeiten. Damit könnte das System an sich recht komfortabel an Ihre Bedürfnisse angepaßt werden. Leider unterstützt Coyote Linux nur die englische Tastatur (QWERTY), weswegen wir im folgenden einen anderen Weg kennenlernen werden, um die Konfiguration des Systems zu ändern.

2) erlaubt es, ein Paßwort für root festzulegen. Damit dieses nach dem nächsten Reboot nicht verlorengeht, muß es dann aber noch zurück auf die Diskette kopiert werden. Dies geschieht mit w). Dabei werden alle wichtigen Konfigurationsdateien in ein Archiv gepackt und auf die Diskette kopiert.

Sobald dieses Paßwort gesetzt ist, kann man sich auch von anderen Rechnern aus mittels SSH auf dem Rechner anmelden. Der Vorteil dabei besteht unter anderem darin, daß man so eine deutsche Tastatur hat und nicht ständig raten muß, welche Tasten den gewünschten Effekt haben.

Mit q) können wir nun das Programm verlassen. Sollten wir es wieder aufrufen wollen, so geschieht das mit dem Befehl:

```
# menu
```

Schauen wir uns nun ein wenig im System um. Schon ein Blick auf die geladenen Module zeigt das Bedürfnis des Autors, jedes nur denkbare Protokoll zu unterstützen. Sämtliche Masquerading-Module des verwendeten Kerns stehen zur Verfügung. Sei es FTP, Chat-Protokolle (IRC, ICQ), Multimedia (RealAudio, H.323) oder Quake, alles ist vorhanden:

```
# lsmod
Module                Size  Used by
ne                     5636   2
8390                   5220   0 [ne]
sch_ingress            1184   0 (unused)
sch_sfq                2752   0 (unused)
sch_htb                17152  0 (unused)
cls_u32                4024   0 (unused)
cls_fw                 2100   0 (unused)
ppp_async              6316   1
ppp_synctty           4984   0 (unused)
ppp_generic            13520  3 [ppp_async ppp_synctty]
slhc                   4164   0 [ppp_generic]
ip_nat_quake3         1768   0 (unused)
```

```

ip_nat_mms          2576  0 (unused)
ip_nat_h323         2028  0 (unused)
ip_nat_amanda       716   0 (unused)
ip_nat_irc          1904  0 (unused)
ip_nat_ftp          2384  0 (unused)
ip_contrack_quake3 1848  1
ip_contrack_mms     2672  1
ip_contrack_h323    2033  1
ip_contrack_egg     2280  0 (unused)
ip_contrack_amanda 1104  1
ip_contrack_irc     2672  1
ip_contrack_ftp     3376  1
softdog             1520  1
    
```

Firewallregeln sind nur wenige definiert. Sie sollten aber für einen Grundschutz ausreichen. Dank Stateful Packet Filtering reichen jeweils drei Regeln für eingehende bzw. weitervermittelte Pakete, um zu verhindern, daß Verbindungen aus dem Internet auf die Firewall bzw. in das lokale Netz möglich sind. Ausgenommen sind hiervon allerdings Verbindungen zum SSH-Server. Diese werden in einer eigenen Regel explizit erlaubt:

```

#iptables -L -v -n
Chain INPUT (policy ACCEPT 1 packets, 58 bytes)
  pkts bytes target     prot opt in     out   source      destination
    99 6414 remote-admin all  --  *     *     0.0.0.0/0  0.0.0.0/0
    37 3292 ACCEPT     all  --  *     *     0.0.0.0/0  0.0.0.0/0
        state RELATED,ESTABLISHED
     0  0 DROP       all  --  *     *     0.0.0.0/0  0.0.0.0/0
        state INVALID
    61 3064 DROP       all  --  ppp0  *     0.0.0.0/0  0.0.0.0/0
        state NEW

Chain FORWARD (policy ACCEPT 124 packets, 7465 bytes)
  pkts bytes target     prot opt in     out   source      destination
   411 113K access-acl all  --  *     *     0.0.0.0/0  0.0.0.0/0
   411 113K autofw-acl all  --  *     *     0.0.0.0/0  0.0.0.0/0
   411 113K portfw-acl all  --  *     *     0.0.0.0/0  0.0.0.0/0
   287 105K ACCEPT     all  --  *     *     0.0.0.0/0  0.0.0.0/0
        state RELATED,ESTABLISHED
     0  0 DROP       all  --  *     *     0.0.0.0/0  0.0.0.0/0
        state INVALID
     0  0 DROP       all  --  ppp0  *     0.0.0.0/0  0.0.0.0/0
        state NEW

Chain OUTPUT (policy ACCEPT 51 packets, 6463 bytes)
  pkts bytes target     prot opt in     out   source      destination

Chain access-acl (1 references)
  pkts bytes target     prot opt in     out   source      destination
    
```

```
Chain autofw-acl (1 references)
pkts bytes target      prot opt in      out source      destination

Chain portfw-acl (1 references)
pkts bytes target      prot opt in      out source      destination

Chain remote-admin (1 references)
pkts bytes target      prot opt in      out source      destination
   0    0 ACCEPT      tcp  --  ppp0  *    0.0.0.0/0  0.0.0.0/0
      tcp spts:1024:65535 dpt:22
```

Als nächstes sehen wir einmal nach, welche Serverdienste auf der Firewall aktiv sind. Da Coyote Linux das Kommando `netstat` nicht enthält, müssen wir dies auf die harte Tour tun. `netstat` interpretiert die Dateien `tcp`, `udp`, `raw` und `unix` im Verzeichnis `/proc/net`. Von denen sind für uns aber im Moment nur die ersten beiden wirklich interessant:

```
# more /proc/net/tcp
sl local_address rem_address  st tx_queue rx_queue tr tm->when
retrnsmt uid timeout inode
 0: 0100A8C0:1FF4 00000000:0000 0A 00000000:00000000 00:00000000
00000000      0 968 1 c3fea040 300 0 0 2 -1
 1: 00000000:0016 00000000:0000 0A 00000000:00000000 00:00000000
00000000      0 946 1 c3db9050 300 0 0 2 -1

# more /proc/net/udp
sl local_address rem_address  st tx_queue rx_queue tr tm->when
retrnsmt uid timeout inode
```

Wir sehen hier zwei TCP-basierte Dienste. Beachten Sie vor allem die zweite und dritte Spalte, die jeweils die lokal verwendete Adresse und den dazugehörigen Port bzw. das Gegenstück auf dem entfernten Rechner in hexadezimaler Darstellung angeben. Da hier noch keine Verbindung zustande gekommen ist, sind Zieladresse und -port 0.

Der erste Dienst ist an die lokale Adresse 0100A8C0 (192.168.0.1²) und den Port 1FF4 (8180) gebunden. Hierbei handelt es sich um das Webinterface. Da es an eine konkrete Adresse gebunden ist, nimmt es nur Pakete von Rechnern an, die sich im selben Subnetz wie diese Adresse befinden. Das bedeutet, der Webserver kann nur aus dem lokalen Netz genutzt werden.

Der zweite Dienst ist dagegen an die Adresse 0 gebunden. Daher werden Verbindungen von jeder Adresse angenommen. Ohne zusätzliche Filterung durch die Firewall kann dieser Dienst also auch aus dem Internet heraus genutzt werden. Als Port verwendet er 0016 (23), den SSH-Port. Es handelt sich also wohl um einen SSH-Server.

Ein Blick auf die aktiven Prozesse ergibt schließlich das folgende Bild:

² Bei Adressen wird das niedrigstwertige Byte zuerst angegeben.

```
# ps
  PID  Uid  VmSize  Stat  Command
    1  root    280  S    init
    2  root      SW    [keventd]
    3  root     SWN   [ksoftirqd_CPU0]
    4  root      SW    [kswapd]
    5  root      SW    [bdflush]
    6  root      SW    [kupdated]
  332  root    204  S    /sbin/watchdog -t 30 /dev/watchdog
  439  root    268  S    syslogd -m 0 -C
  443  root    240  S    klogd
  450  root    280  S    /usr/sbin/sshd -p 22
  548  root    356  S    -sh
  549  root    252  S    /sbin/getty 38400 tty2
  550  root    252  S    /sbin/getty 38400 tty3
  551  root    364  S    /usr/sbin/tlhttpd -u root -d /var/http/htdocs -nor -h
  646  root    272  S    sh -c pppoe -I eth1 -m 1412
  647  root    200  S    pppoe -I eth1 -m 1412
  650  root    412  S    pppd
 1260  root    260  R    ps
 1261  root    356  R    -sh
```

Interessant sind hierbei vor allem:

/usr/sbin/sshd Der SSH-Dienst, der es erlaubt, sich über das Netz anzumelden, als säße man direkt vor dem Rechner.

/usr/sbin/tlhttpd Der Webserver, der eine Konfigurationsoberfläche zur Verfügung stellt.

pppd stellt die Internet-Verbindung über das Modem her.

pppoe Der Beweis, daß ich geschummelt habe und hier die Prozeßliste eines DSL-Systems abdrucke. pppoe wird vom pppd für den DSL-Verbindungsaufbau genutzt.

Die Konfiguration anpassen

Grundsätzlich ist Coyote Linux schon recht nahe an den Bedürfnissen unseres Szenarios. Es hat nur zwei Schönheitsfehler in Gestalt der beiden aktiven Netzwerkdienste, von denen einer in der Standardeinstellung sogar aus dem Internet zugänglich ist.

Die einfachste Möglichkeit, diese zumindest teilweise zu beheben, besteht darin, im Webinterface das Menü *Administrative Configs* auszuwählen. Der Unterpunkt *Enable External SSH Access* bestimmt, ob eine Firewallregel aktiviert ist, die den externen Zugriff auf den SSH-Server erlaubt. Der Punkt *Enable Web Administrator* legt dagegen fest, ob der Webserver gestartet wird. Beide Einstellungen werden erst aktiv, wenn man seine Einstellungen auf Diskette abspeichert und neu bootet.

Sieht man sich nach dieser Aktion einmal die Hauptkonfigurationsdatei */config/coyote.cfg* auf der Diskette an, so findet man dort einige neue Variablen, die vorher nicht vorhanden waren. Interessant sind dabei insbesondere die folgenden beiden:

```
ENABLE_EXTERNAL_SSH='NO'
ENABLE_WEBADMIN='NO'
```

Die erste Variable steuert, ob die Firewall externe Verbindungen zum SSH-Server zuläßt, die zweite, ob der Webserver gestartet wird.

Wollen wir diese Variablen ohne Zuhilfenahme des Webinterfaces ändern, so haben wir zwei Möglichkeiten. Wir können im Menü den Punkt 1) auswählen. Dann wird ein Editor gestartet, mit dem wir die Konfigurationsdatei bearbeiten können. Wir sollten dies allerdings möglichst über eine SSH-Verbindung von einem anderen Rechner aus machen. Arbeiten wir lokal am Rechner, so müssen wir uns mit einer amerikanischen Tastaturbelegung abfinden. Wir sollten auch nicht vergessen, die Änderungen mit w) auf die Diskette zu schreiben, wenn wir fertig sind. Andernfalls hätten wir nur eine Kopie der Datei im Hauptspeicher verändert.

Eine Alternative besteht darin, die Diskette in einem normalen Linuxsystem zu mounten und die Konfigurationsdatei ganz normal mit einem Editor zu öffnen. Dies hat den Vorteil, daß man seinen Lieblingseditor benutzen kann.

Beim Mounten der Diskette dürfen wir aber nicht, wie normal üblich, das Device `/dev/fd0` verwenden, da es sich um eine hochformatierte Diskette handelt, `/dev/fd0` sie aber als normale 1,44-MB-Floppy anspricht. Wir würden so nur sinnlosen Datenmüll lesen. Statt dessen verwenden wir `/dev/fd0u1680`, wenn wir die Diskette wie oben vorgeschlagen auf 1,68 MB formatiert haben:

```
# mount /dev/fd0u1680 /floppy -tvfat
```

Nun können wir ganz normal in das Verzeichnis `/floppy/config` wechseln und die Datei `coyote.cfg` editieren.

Ein Problem bleibt allerdings. Wir können auf diese Weise den SSH-Server nicht völlig deaktivieren. Wollen wir dies tun, so müssen wir tiefer vordringen. Die Diskette enthält eine Reihe von `.tgz`-Archiven, die zur Laufzeit entpackt und installiert werden. In `etc.tgz` befinden sich dabei einige Runlevel-Skripte³, die unter anderem auch für den Start des SSH-Dienstes sorgen. Durch Editieren dieser Dateien können wir dafür sorgen, daß schließlich keine Netzwerkdienste mehr aktiv sind.

Dazu sollten wir in ein neu angelegtes Verzeichnis wechseln, das Archiv von der Diskette herüberkopieren und es auspacken:

```
# mkdir /tmp/coyote.etc
# cd /tmp/coyote.etc
# cp /floppy/etc.tgz .
# tar xvzf etc.tgz
```

Nachdem es entpackt ist, brauchen wir das ursprüngliche Archiv nicht mehr:

```
# rm etc.tgz
```

³ Vgl. Kapitel 8, Unterabschnitt *Init*, ab Seite 125

Im Unterverzeichnis *etc/rc.d/* finden wir nun die Runlevel-Skripte:

```
# cd etc/rc.d
# ls
pkgs rc.dhcp      rc.inet      rc.masquerade rc.qos
rc.M rc.dnsmasq    rc.line_up   rc.ppp         rc.qos.coyote
rc.S rc.firewall   rc.local     rc.pppoe       rc.qos.wonder
```

Die für uns relevante Datei ist *rc.inet*. In ihr finden wir den folgenden Abschnitt:

```
if [ "$LOCAL_UP" = ÜP" ]; then
    echo "Starting SSH daemon..."
    if [ -z "$SSH_PORT" ] ; then SSH_PORT=22 ; fi
    /usr/sbin/sshd -p $SSH_PORT

    /etc/rc.d/rc.dnsmasq
fi
```

Wir brauchen nur den Aufruf des *sshd* auszukommentieren, und der Dienst wird nicht gestartet:

```
if [ "$LOCAL_UP" = ÜP" ]; then
    echo "Starting SSH daemon..."
    if [ -z "$SSH_PORT" ] ; then SSH_PORT=22 ; fi

    # -- we do not want sshd -----
    # /usr/sbin/sshd -p $SSH_PORT
    # -----

    /etc/rc.d/rc.dnsmasq
fi
```

Allerdings dürfen wir nicht vergessen, das Archiv wieder zusammenzubauen:

```
# cd /tmp/coyote.etc
# tar cvf etc.tar
# gzip -9 etc.tar
# mv etc.tar.gz etc.tgz
```

Vielleicht wundern Sie sich, daß ich hier *tar* nicht einfach den Parameter *-z* übergebe, sondern explizit *gzip* aufrufe. Der Unterschied ist, daß der Aufruf von *gzip* mit dem Parameter *-9* besonders stark komprimiert. Da wir nur eine Diskette zur Verfügung haben, ist es sinnvoll, mit dem zur Verfügung stehenden Platz sparsam umzugehen.

Schließlich müssen wir das Archiv nur noch zurück auf die Diskette kopieren:

```
# rm /floppy/etc.tgz
# cp etc.tgz /floppy/
# umount /floppy
```

fli4l

fli4l verfolgt einen etwas anderen Ansatz als Coyote. In der Standardkonfiguration ist es deutlich spartanischer. Allerdings existiert eine ganze Reihe von Zusatzpaketen, mit denen man eine Vielzahl von Funktionen nachrüsten kann. Nutzt man dies voll aus, so übersteigt der Funktionsumfang von fli4l den von Coyote deutlich. Für unsere Zwecke reicht aber eine Grundkonfiguration völlig aus.

fli4l unterstützt im Gegensatz zu Coyote auch ISDN. Allerdings werden in der momentan aktuellen Version (2.0.8) nur ISDN-Karten unterstützt, die kaum noch zu bekommen sind. So wird zwar die alte Fritz!Card PCI unterstützt, im Laden ist aber nur die Fritz!Card PCI v2 zu bekommen. Diese zweite Version der Karte ist leider nicht hinreichend kompatibel zu ihrer Vorgängerversion, daß deren Treiber sie unterstützen würde.

Wollen Sie also ISDN nutzen, müssen Sie gegenwärtig die Entwicklungsversion von fli4l (2.1.x) verwenden. Diese ändert sich derzeit aber noch erheblich von Version zu Version. Ich werde sie hier daher nicht beschreiben. Jede Darstellung wäre wahrscheinlich schon veraltet, bevor dieses Buch gedruckt wird. Immerhin unterstützt sie die Fritz!Card PCI v2 unter der HiSax-Nummer 39, auch wenn man nach dieser Information tief im FAQ wühlen muß.

Konfiguration

Sie können fli4l von folgender Adresse herunterladen:

```
http://www.fli4l.de
```

Im Bereich »Download« finden Sie eine ganze Reihe von Paketen, welche die unterschiedlichsten Zusatzfunktionen enthalten. Für eine einfache Firewall mit ISDN- oder DSL-Unterstützung benötigen Sie aber nur die Dateien

- *fli4l-<Version>.tar.gz*
- *isdn.tar.gz* bzw. *dsl.tar.gz*

Entpacken Sie nun das Basisarchiv (hier *fli4l-2.0.8.tar.gz*):

```
# tar xvzf fli4l-2.0.8.tar.gz
```

Damit ist ein neues Unterverzeichnis entstanden (hier: *fli4l-2.0.8*). Eventuell gewünschte Zusatzpakete müssen relativ zu diesem Verzeichnis ausgepackt werden. Man kann z. B. tar mit der Option -C mitteilen, erst in das Verzeichnis zu wechseln, bevor es mit dem Auspacken beginnt:

```
# tar xvzf isdn.tar.gz -C fli4l-2.0.8
```

Haben wir alle Pakete ausgepackt, so können wir nun selbst in das Verzeichnis wechseln:

```
# cd fli4l-2.0.8
```

Im Unterverzeichnis *config/* finden Sie die Konfigurationsdateien der einzelnen Pakete, deren Inhalte Sie an Ihre konkrete Situation anpassen müssen. Beginnen wir mit dem Grundpaket:

```
# vi config/base.txt
```

Die Datei enthält diverse Variablenzuweisungen der Art

```
<Variable>=<Wert>
```

Am Anfang stehen einige Grundeinstellungen wie z. B.:

HOSTNAME Name unserer Firewall.

PASSWORD Paßwort für die Anmeldung als root sowohl lokal als auch über Telnet, FTP und SSH (falls vorhanden). Hier sollten Sie unbedingt ein sicheres Paßwort eintragen.

MOUNT_BOOT gibt an, ob die Diskette nach dem Systemstart auf */boot* gemountet werden soll. Im Normalfall können Sie hier *no* eintragen. Die Diskette wird dann nicht gemountet und kann nach dem Bootvorgang aus dem Rechner entfernt werden.

Eine Reihe von Variablen definiert die Konfiguration der Netzwerkkarten:

ETH_DRV_N Anzahl der vorhandenen Netzwerkkarten. Für ISDN ist dieser Wert 1. Für DSL ist er 2, wenn zwei verschiedene Netzwerkkarten verwendet werden. Verwendet man dagegen zwei baugleiche Netzwerkkarten, so kann hier 1 angegeben werden.

ETH_DRV_1 Name des Treibers für die erste Netzwerkkarte (2. Karte: *ETH_DRV_2*). Dabei ist bei einigen Treibern zu beachten, daß diese einen weiteren Treiber benötigen, der zuerst geladen werden muß. Ein Beispiel ist *ne*, der Treiber für NE2000-Netzwerkkarten. Dieser benötigt den Treiber 8390. Sie müssen in die Variable also "8390 ne" eintragen. Weitere Beispiele finden Sie z. B. in *doc/deutsch/text/readme.txt*.⁴ Bei DSL ist hier als erste Karte diejenige anzugeben, die mit dem lokalen Netz verbunden ist. Als zweite nehmen Sie bitte diejenige, die an das DSL-Modem angeschlossen ist.

ETH_DRV_<n>_OPTION Parameter für den Treiber. Bei ISA-Karten sollte man hier zumindest den verwendeten Port angeben (z. B. "io=0x340"). Verwendet man zwei baugleiche Karten, so gibt man die Parameter mit Komma getrennt an (z. B. "io=0x320,0x300").

IP_ETH_N Anzahl der Netzwerkkarten, denen eine IP-Adresse zugewiesen werden soll (normalerweise: 1). Bei DSL wird der Netzwerkkarte, die mit dem DSL-Modem verbunden ist, keine IP-Adresse zugewiesen. Diese Karte zählt hier also nicht.

IP_ETH_1_NAME Device-Name, mit dem die erste (s. o.) Netzwerkkarte angesprochen werden kann. Kann entfallen, wenn es sich um eine Ethernet-Karte handelt (*eth0*).

⁴ In Version 2.0.8 war dies offensichtlich nicht mehr nötig. *fi4l* funktionierte auch problemlos, wenn ich nur "ne" angegeben habe.

IP_ETH_1_IPADDR Die zugehörige IP-Adresse (z. B. 192.168.20.1).

IP_ETH_1_NETWORK Das zugehörige Netz (z. B. 192.168.20.0).

IP_ETH_1_NETMASK Die zugehörige Netzmaske (z. B. 255.255.255.0).

Als nächstes folgen Einstellungen zum Forwarding und Masquerading. Hier sind die Vorgaben durchaus sinnvoll gewählt, man sollte aber sicherheitshalber überprüfen, ob sie die eigene Situation wirklich abdecken:

MASQ_NETWORK bezeichnet das lokale Netz. Die Angabe sollte zu einem der zuvor definierten Netze passen (z. B. 192.168.20/24)⁵.

MASQ_MODULE_N Anzahl der Masquerading-Module, die geladen werden sollen.

MASQ_MODULE_<n> definiert ein Masquerading-Modul. Ist <n> kleiner oder gleich **MASQ_MODULE_N**, so wird das Modul beim Hochfahren automatisch geladen. Hier ist eine Reihe von Modulen vordefiniert, von denen aber standardmäßig nur das erste (ftp) aktiviert ist.

FORWARD_DENY_HOST_N Anzahl der Rechner, die auf keinen Fall Kontakt mit dem Internet haben dürfen und deshalb Spezialregeln benötigen.

FORWARD_DENY_HOST_<n> definiert einen Rechner. Ist <n> kleiner oder gleich **FORWARD_DENY_HOST_N**, so werden Pakete dieses Rechners nicht in andere Netze weitergeleitet.

FORWARD_DENY_PORT_N entspricht **FORWARD_DENY_HOST_N**, dient aber dazu, Ports zu definieren, nicht Rechneradressen.

FORWARD_DENY_PORT_<n> entspricht **FORWARD_DENY_HOST_<n>**, dient aber der Definition von Ports, nicht von Rechneradressen. Die Angabe erfolgt hier in Form eines Portbereiches gefolgt von einer Ablehnungsart (DENY/REJECT), z. B.:

```
'137:139 REJECT'
```

Diese Definition ist standardmäßig vorhanden und sorgt dafür, daß der Zugriff auf Netzwerkfreigaben im Internet unterbunden wird. Da diese Zugriffe regelmäßig normale Zugriffe z. B. auf Web- oder FTP-Server begleiten, ist es sinnvoll, sie explizit zu verbieten.

Ob man im Einzelfall DENY oder REJECT wählt, ist eine Geschmacksfrage. REJECT sendet eine Fehlermeldung, wie sie auch erfolgt wäre, wenn auf dem Zielsystem der betreffende Dienst nicht installiert ist. REJECT bietet sich vor allem in Fällen an, wo eine Anfrage zwar unterbunden werden soll, die Anfrage aber keinen Angriff darstellt, sondern grundsätzlich legitim ist.

Verwendet man DENY, so wird das betreffende Paket entsorgt, ohne daß eine Fehlermeldung gesendet wird. Der Anfragende wird im unklaren gelassen und wartet unter Umständen eine ganze Weile auf Antwort. Dies ist insbesondere bei Zugriffen sinnvoll, die man als Angriff oder Vorform dazu ansieht. Führt der Angreifer

⁵ Hier wird eine andere Methode benutzt, um zu definieren, welche Bits einer Adresse das Netzwerk und welche den Rechner angeben. Die 24 bedeutet, daß die ersten 24 Bits das Netzwerk angeben. Dies ist gleichbedeutend mit der Maske 255.255.255.0 (Binär: 11111111.11111111.11111111.00000000).

z. B. einen Port Scan durch, so können DENY-Regeln sein Vorgehen deutlich verlangsamen. Je nach Programmierung des von ihm verwendeten Programms wird es womöglich nach jedem Versuch, einen Port zu untersuchen, erst einmal eine Pause einlegen.

Ging es gerade darum, welche Pakete nicht in das Internet weitergeleitet werden sollten, so wird nun definiert, welche Pakete aus dem Internet nicht in das lokale Netz gelangen dürfen. Auch hier sind die Regeln prinzipiell brauchbar gewählt, man sollte aber trotzdem ein paar Minuten darüber nachdenken, was sie bedeuten:

FIREWALL_DENY_PORT_N Anzahl der Regeln, die festlegen, welche Ports für den Zugriff aus dem Internet gesperrt sind.

FIREWALL_DENY_PORT_<n> enthält einen Bereich von Ports, die aus dem Internet nicht zu erreichen sein dürfen. Damit die Regel auch beachtet wird, muß **FIREWALL_DENY_PORT_N** größer oder gleich <n> sein. Die Angabe erfolgt hier in Form eines Portbereiches gefolgt von einer Ablehnungsart (DENY/REJECT) wie zum Beispiel:

```
'0:52 REJECT'
```

Alle hier definierten Ports werden sowohl für eingehende UDP-Pakete als auch für eingehende TCP-Verbindungsanfragen gesperrt. Eingehende TCP-Antwortpakete werden dagegen immer zugelassen.

Standardmäßig sind hier die Ports unter 1024 mit Ausnahme der Ports 53 (DNS) und 113 (Ident) sowie die Ports 5000 (imond), 5001 (telnetd), 8000 (junkbuster) und 20012 (Vbox Server) eingetragen. Der Zugriff auf Port 53 muß erlaubt werden, damit die Namensauflösung funktioniert.

Üblicherweise wird auf einer fli4l-Firewall auch ein DNS-Server betrieben. Sind nämlich Windows-Rechner so konfiguriert, daß sie einen DNS-Server benutzen, so werden sie auch bei dem Versuch, Namen im lokalen Netz aufzulösen, DNS-Anfragen stellen. Befindet sich der eingetragene DNS-Server im Internet, so kann dies teuer werden, da ständig Verbindungen zum Internet aufgebaut werden.

Abhilfe schafft hier der DNS-Server auf der Firewall. Kennt er die Rechner im lokalen Netz, so kann er die Anfragen beantworten, ohne dafür eine Verbindung zum Internet aufzubauen. Dazu ist es nötig, daß Sie die Klienten in Ihrem lokalen Netz so konfigurieren, daß sie als DNS-Server die Firewall nutzen. Außerdem müssen Sie als DNS-Domäne dieselbe Domäne eintragen, die Ihr DNS-Server benutzt.

Umgekehrt müssen Sie auch den DNS-Server der Firewall so konfigurieren, daß er sich für Ihre private Domäne zuständig fühlt, und ihm die Namen der Rechner in Ihrem lokalen Netz beibringen. Ihre Rechner sollten dabei Namen haben, die nur aus Buchstaben, Zahlen und Minuszeichen bestehen.

START_DNS gibt an, ob der DNS-Server aktiv sein soll. Unbedingt auf 'yes' lassen.

DNS_FORWARDERS Hier werden die IP-Adressen der DNS-Server eingetragen, bei denen der Server auf der Firewall nachfragen kann, wenn er selbst die Antwort nicht kennt. Normalerweise handelt es sich dabei um den oder die Server Ihres Providers. Sollen mehrere Server benutzt werden, so sind die einzelnen Adressen durch Leerzeichen zu trennen.

DOMAIN_NAME Ihre private Domäne für das lokale Netz, z. B. 'nowhere.invalid'.

HOSTS_N Anzahl der DNS-Einträge. Idealerweise ist dies die Anzahl der Rechner im lokalen Netz plus einem Eintrag für das interne Interface der Firewall.

HOST_<n> Die einzelnen Einträge. <n> kann dabei Werte zwischen 1 und **HOSTS_N** annehmen. Ein Eintrag besteht aus einer IP-Adresse gefolgt von einem Rechnernamen, z. B. '192.168.6.1 fli41'.

Benutzt man ISDN oder DSL, so verlangt das Installationsprogramm, daß man auch den **imond** aktiviert. Dieses Programm hat zwei Funktionen. Zum einen hat der **imond** dafür zu sorgen, daß zu verschiedenen Uhrzeiten verschiedene Provider für die Einwahl benutzt werden. Zum anderen erlaubt er es, den Rechner über das Netz fernzusteuern. Zugriffe aus dem Internet werden aber normalerweise von den voreingestellten Firewallregeln abgefangen.

Hätte ich die Wahl, so würde ich ihn wahrscheinlich nicht aktivieren. Zum einen wäre es mir zuviel Aufwand, die Preislisten für die Auswahl der Provider aktuell zu halten, zum anderen stellt jeder Serverdienst auf einer Firewall ein nicht zu unterschätzendes Risiko dar. Darüber hinaus sind unsere »Kunden« normale Anwender, die nicht die Firewall administrieren, sondern lediglich surfen wollen. Wir selbst dagegen kennen das Root-Paßwort und können den Server lokal administrieren, indem wir uns einfach davorsetzen. Es besteht daher überhaupt keine Notwendigkeit für einen Zugriff über das Netz.

Solange die Firewall so konfiguriert ist, daß sie automatisch den Provider auswählt, wenn Pakete eintreffen, die für das Internet bestimmt sind, reicht es, sie einmal in eine Ecke zu stellen und zu starten. Die Anwender können dann jederzeit problemlos auf das Internet zugreifen, ohne überhaupt darüber nachzudenken, daß sie eine Firewall benutzen.

START_IMOND erlaubt den Start des Dienstes. Bei der Nutzung von ISDN oder DSL muß hier 'yes' eingetragen sein.

IMOND_PASS legt das Paßwort fest, um mit normalen Anwenderrechten auf den **imond** zuzugreifen. Wenn es leer ist, nimmt der **imond** von jedermann Befehle entgegen. Damit können zumindest diverse statistische Daten zur Verbindung, die CPU-Auslastung, die aktuelle Zeit und Protokolldateien abgerufen werden. Einige weitergehende Kommandos können für normale Benutzer freigeschaltet werden.

Vergeben Sie hier ein starkes Paßwort. Wenn wir schon einen Server auf der Firewall betreiben, so sollte er wenigstens sicher konfiguriert sein.

IMOND_ADMIN_PASS legt das Paßwort fest, das eingegeben werden muß, um mit Administratorrechten auf den **imond** zuzugreifen. Wenn es leer ist, reicht die Eingabe des Anwenderpaßwortes.

Zu den Befehlen, die dem Administrator vorbehalten sind, gehört unter anderem das Einschalten von Channel-Bundling, das Löschen von Protokolldateien und die Übertragung von beliebigen Dateien zwischen der eigenen Arbeitsstation und der Firewall.

Dieses Paßwort hat denselben Stellenwert wie das Root-Paßwort. Wählen Sie ein sicheres Paßwort, und teilen Sie es nur Personen mit, denen Sie auch das Root-Paßwort geben würden.

IMOND_LOG Wird diese Variable auf 'yes' gesetzt, so werden die einzelnen Verbindungen mit Provider, Datum, Uhrzeit, Dauer und Kosten in der Datei *imond.log* protokolliert.

Dies ist nur dann sinnvoll, wenn Sie Ihre Telefonrechnung überprüfen wollen. Wenn Sie nicht wirklich vorhaben, das entstehende Protokoll auch auszuwerten, sollten Sie darauf verzichten, den knappen Speicherplatz zu vergeuden. Im Normalfall kann man hier also 'no' eintragen.

IMOND_LOGDIR gibt das Verzeichnis für die Protokolldatei des imond an. Üblich ist */var/log/*.

IMOND_ENABLE legt fest, ob Benutzer die automatische Einwahl ins Internet an- und abschalten dürfen.

In unserem Szenario ist dies nicht sinnvoll. Dies führt nur dazu, daß ein Anwender die Einwahl ins Internet abschaltet, worauf sich dann ein anderer wundert, warum er nicht surfen kann. Eine längere Fehlersuche ist vorprogrammiert. Wählen Sie daher 'no'.

IMOND_DIAL gibt an, ob normale Benutzer eine manuelle Einwahl auslösen und einen Befehl zum Auflegen geben dürfen.

Wenn wir eine automatische Einwahl konfiguriert haben, so ist es nicht nötig, manuell zu wählen. Tragen Sie daher hier 'no' ein.

IMOND_ROUTE bestimmt, ob ein normaler Benutzer die Firewall anweisen kann, welchen der vorkonfigurierten Provider sie benutzen soll.

Dies ist etwas, womit sich normale Nutzer nicht befassen wollen. Es ist Ihr Job zu entscheiden, welche Provider genutzt werden sollen. Wählen Sie daher 'no'.

IMOND_REBOOT erlaubt einem Benutzer, die Firewall herunterzufahren oder einen Neustart auszulösen.

Wenn Sie davon ausgehen, daß die Firewall einmal eingeschaltet permanent aktiv bleibt, so können Sie getrost 'no' eintragen. Sollen die Anwender die Firewall aber nur bei Bedarf starten und nach Abschluß ihres Ausflugs in das Internet wieder sauber herunterfahren, so kann es sinnvoll sein, hier 'yes' einzutragen.

Um die allgemeine Netzwerkkonfiguration abzurunden, folgen noch zwei Grundeinstellungen:

IP_DYN_ADDR gibt an, ob die IP-Adresse vom Provider dynamisch vergeben wird.

Wenn Sie nicht explizit eine statische Adresse vereinbart haben, so ist 'yes' die richtige Einstellung.

DIALMODE legt fest, ob eine Einwahl ins Internet automatisch bei Bedarf ('auto'), auf ausdrückliche Anforderung z. B. mit dem imond ('manual') oder überhaupt nicht ('off') erfolgen soll.

Im Normalfall werden Sie hier 'auto' eintragen.

Schließlich bleibt noch die Protokollierung wichtiger Ereignisse. Standardmäßig ist diese nicht aktiviert. Da unser System nur eine kleine RAM-Disk zum Speichern seiner Protokolle zur Verfügung hat, bestünde beim Speichern in Form von Dateien die Gefahr, in kürzester Zeit allen verfügbaren Platz aufzubrechen. Andererseits sind die Meldungen im Systemprotokoll oft eine gute Möglichkeit herauszufinden, warum etwas nicht so funktioniert, wie es sollte. Auch will man es unter Umständen sehen, falls unerlaubte Zugriffe aus dem Internet erfolgen.

Eine Alternative zur Protokollierung in lokalen Dateien könnte hier darin bestehen, einen Rechner im lokalen Netz so zu konfigurieren, daß er Protokollmeldungen über das Netz annimmt. Die Firewall kann ihm dann ihre Meldungen schicken. Eine andere Lösung besteht darin, die Protokollmeldungen auf dem Bildschirm anzeigen zu lassen. Dort werden sie zwar schnell von neuen Meldungen verdrängt, man hat aber zumindest die Chance festzustellen, was gerade aktuell passiert.

Wollen Sie also eine Protokollierung wichtiger Ereignisse, so müssen Sie die folgenden Variablen anpassen:

OPT_SYSLOGD bestimmt, ob eine Protokollierung erfolgt. Hier sollte also 'yes' stehen.

SYSLOGD_DEST_N gibt die Anzahl der Regeln an, die spezifizieren, wohin Protokolle geschrieben werden.

SYSLOGD_DEST_<n> enthält die <n>te Regel. Regeln bestehen aus einem Muster, das angibt, welche Meldungen protokolliert und wohin die Meldungen geschrieben werden sollen. Eine detailliertere Darstellung finden Sie in Kapitel 9, Abschnitt *Das Systemprotokoll*, ab Seite 205. Für unsere Zwecke reicht aber als Muster *.* , womit alle Meldungen protokolliert werden. Als Ziel kann z. B. eine Datei (*/var/log/messages*), eine Konsole (*/dev/tty9*) oder ein Rechner im lokalen Netz (*@192.168.20.15*) angegeben werden.

OPT_KLOGD bewirkt, daß Fehlermeldungen des Kernels (z. B. verbotene Zugriffe aus dem Internet) an den *syslogd* zur Protokollierung weitergegeben werden. Die Variable sollte daher auf 'yes' gesetzt werden, wenn eine Protokollierung konfiguriert wurde.

Wollen wir z. B. sowohl auf Konsole 9 als auch auf einen Rechner im Netz protokollieren, so brauchen wir die folgenden Einträge:

```
OPT_SYSLOGD='yes'  
SYSLOGD_DEST_N='2'  
SYSLOGD_DEST_1='*.* /dev/tty9'  
SYSLOGD_DEST_2='*.* @192.168.20.15'  
OPT_KLOGD='yes'
```

Damit hätten wir den allgemeinen Teil der Konfiguration abgeschlossen. Wir können die Datei nun speichern und verlassen.

Wir sind allerdings noch nicht fertig. Die Konfiguration von ISDN bzw. DSL erfolgt in einer eigenen Konfigurationsdatei.

Der Zugang über DSL wird in der Datei *config/dsl.txt* konfiguriert. Dabei reicht es im Allgemeinen, die folgenden Variablen anzupassen:

OPT_PPPOE Hier muß unbedingt *yes* eingetragen sein, damit die Einwahl mittels DSL erfolgt.

PPPOE_USEPEERDNS sorgt dafür, daß automatisch der DNS-Server benutzt wird, den der Provider bei der Anmeldung mitteilt.

PPPOE_ETH gibt das Netzwerk-Interface an, das an das DSL-Modem angeschlossen ist (normalerweise: *eth1*).

PPPOE_USER legt den Benutzernamen fest.

PPPOE_PASS enthält das Paßwort.

PPPOE_HUP_TIMEOUT bewirkt, daß die Verbindung beendet wird, wenn die angegebene Anzahl von Sekunden keine Daten übertragen werden.

PPPOE_CHARGEINT legt das Abrechnungsintervall des Providers fest. Bei sekunden-genauer Abrechnung sollte hier also *'1'* stehen.

PPPOE_TIMES enthält durch Leerzeichen getrennte Einträge der Art

Mo-Su:00-24:0.0148:Y

Damit wird für einen Bereich von Wochentagen und Uhrzeiten ein Preis festgelegt. Endet der Eintrag auf *N*, so darf der Provider zu der festgelegten Zeit nicht benutzt werden.

Der Zugang über ISDN wird dagegen in *config/isdn.txt* konfiguriert. Wir beginnen damit, daß wir unsere ISDN-Karte definieren. Dazu dienen die folgenden Variablen:

OPT_ISDN sollte auf *'yes'* gesetzt sein, damit ISDN genutzt wird.

ISDN_TYPE gibt die HiSax-Nummer der Karte an.

ISDN_IO legt die Portadresse fest.

ISDN_IO0, ISDN_IO1 sind zusätzliche Portadressen, die von einigen Karten benötigt werden.

ISDN_MEM definiert eine Basisadresse im Hauptspeicher.

ISDN_IRQ gibt den benutzten Interrupt an.

Bei einer PCI-Karte wie meiner Fritz!Card beschränken sich die Parameter, die man herausfinden muß, in der Regel auf die Nummer, unter der der Treiber HiSax die Karte kennt:

```
OPT_ISDN='yes'  
ISDN_TYPE='27'  
ISDN_IO=''  
ISDN_IO0=''  
ISDN_IO1=''  
ISDN_MEM=''  
ISDN_IRQ=''
```

Nun ist es noch nötig zu definieren, welche Provider man nutzen will. Standardmäßig ist hier MSN vordefiniert. Wenn wir einen oder mehrere andere Provider nutzen wollen, so müssen wir die Definition anpassen bzw. weitere Definitionen einfügen.

Die meisten Variablen kann man aus der Definition für MSN übernehmen. Die folgenden sollte man aber in jedem Fall an den verwendeten Provider anpassen:

ISDN_CIRCUITS_N gibt an, wie viele Provider (Circuits) benutzt werden sollen.

ISDN_CIRC_<n>_USEPEERDNS sorgt dafür, daß automatisch der DNS-Server benutzt wird, den der Provider bei der Anmeldung mitteilt.

ISDN_CIRC_<n>_USER legt den eigenen Benutzernamen fest.

ISDN_CIRC_<n>_PASS gibt das zu verwendende Paßwort an.

ISDN_CIRC_<n>_DIALOUT definiert die Telefonnummer des Providers.

ISDN_CIRC_<n>_EAZ stellt die eigene MSN ein.

Hat man mehrere Provider definiert und möchte nun, daß jeweils der billigste gewählt wird, so müssen auch die folgenden beiden Variablen angepaßt werden:

ISDN_CIRC_<n>_HUP_TIMEOUT bewirkt, daß die Verbindung beendet wird, wenn die angegebene Anzahl von Sekunden keine Daten übertragen werden.

ISDN_CIRC_<n>_CHARGEINT legt das Abrechnungsintervall des Providers fest. Bei sekundengenaue Abrechnung sollte hier also '1' stehen.

ISDN_CIRC_<n>_TIMES enthält durch Leerzeichen getrennte Einträge der Art

```
Mo-Su:00-24:0.0148:Y
```

Damit wird für einen Bereich von Wochentagen und Uhrzeiten ein Preis festgelegt. Endet der Eintrag auf N, so darf der Provider zu der festgelegten Zeit nicht benutzt werden.

Der imond erstellt später zur Laufzeit aus den CHARGEINT-Informationen zu den einzelnen Anbietern eine Tabelle, in der für jede Stunde jeden Tages ein Provider festgelegt ist. Er überprüft dann minütlich, ob gerade der Provider aktiviert ist, der für die aktuelle Uhrzeit am günstigsten ist. Ist dies nicht der Fall, so unterbricht er gegebenenfalls bestehende Verbindungen und konfiguriert das System zur Benutzung des aktuellen Providers um. Zu Zeiten, zu denen kein Provider definiert ist, kann das Internet nicht genutzt werden.

Nun ist die Konfiguration abgeschlossen. Die Diskette kann generiert werden. Dazu legen wir eine leere Diskette ein und rufen das Skript mkfloppy.sh auf:

```
# ./mkfloppy.sh
cc -g -Wall -c -o mkfli4l.o mkfli4l.c
cc -g -Wall -c -o var.o var.c
cc -g -Wall -c -o check.o check.c
cc mkfli4l.o var.o check.o -o mkfli4l
creating compressed tar archive...
[...]
writing FAT and system files ...
copying syslinux.cfg ...
copying kernel ...
copying rootfs.gz ...
copying rc.cfg ...
copying opt.tgz ...
```

Unsere Diskette ist nun fertig. Wir können sie in den Firewallrechner einlegen und von ihr booten. Nehmen Sie die Diskette nach dem Bootvorgang bitte nicht heraus. fli4l mountet sie im Gegensatz zu Coyote Linux. Sie zu entfernen wäre so, als ob Sie eine Festplatte im laufenden Betrieb herausnehmen würden.

Ein erster Rundgang

Im Gegensatz zu Coyote Linux müssen Sie bei fli4l ein Paßwort angeben, wenn Sie sich als root anmelden. Sobald die Anmeldung abgeschlossen ist, finden Sie sich auf einer Kommandozeile wieder. Nicht nur hat fli4l kein Konfigurationsprogramm wie Coyote Linux, es fehlen auch so elementare Befehle wie `more` oder ein Editor. Damit sind unsere Möglichkeiten, die Konfiguration zu überprüfen oder gar zu ändern, stark eingeschränkt. Nicht einmal Dateien können wir uns vernünftig anzeigen lassen, falls diese größer als eine Bildschirmseite sind.

Damit wird schon die Überprüfung der Firewallregeln kompliziert. Ich habe sie daher in eine Datei auf der Systemdiskette ausgegeben, um sie mir dann später auf einem anderen Rechner anzusehen.

```
# mount /dev/fd0 /mnt
# ipchains -L -v -n > /mnt/rules.txt
```

Dies wird allerdings etwas erschwert, da fli4l keine Unterstützung für die deutsche Tastatur mitbringt. Falls Sie die Belegung der amerikanischen Tastatur nicht im Kopf haben, hier ein paar Tips:

Zeichen	Taste
y	z
z	y
-	Ziffernblock —
/	Ziffernblock ÷
*	Ziffernblock *

Die Zeichen `>`, `<`, `|`, `.` und `,` befinden sich dagegen an ihrem gewohnten Platz.

Nachdem ich die Firewall heruntergefahren hatte, nahm ich die Diskette und ging zu einem anderen Linux-System. Hier das Ergebnis, wobei ich jedoch der Übersichtlichkeit halber einige Spalten ausgelassen habe, die für diese Betrachtung unwichtig sind:

```
# mount /dev/fd0 /floppy -tvfat
# less /floppy/rules.txt
Chain input (policy ACCEPT: 0 packets, 0 bytes):
target prot opt ifname source destination ports
ACCEPT all ----- * 127.0.0.1 127.0.0.1 n/a
ACCEPT tcp !y---- * 0.0.0.0/0 0.0.0.0/0 * -> *
ACCEPT all ----- * 192.168.20.0/24 0.0.0.0/0 n/a
REJECT tcp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 0:52
REJECT udp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 0:52
REJECT tcp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 54:112
REJECT udp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 54:112
REJECT tcp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 114:1023
REJECT udp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 114:1023
REJECT tcp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 5000:5001
REJECT udp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 5000:5001
REJECT tcp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 8000
REJECT udp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 8000
REJECT tcp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 20012
REJECT udp ----l- * 0.0.0.0/0 0.0.0.0/0 * -> 20012
Chain forward (policy DENY: 0 packets, 0 bytes):
target prot opt ifname source destination ports
REJECT tcp ----- * 0.0.0.0/0 0.0.0.0/0 * -> 137:139
REJECT tcp ----- * 0.0.0.0/0 0.0.0.0/0 137:139 -> *
REJECT udp ----- * 0.0.0.0/0 0.0.0.0/0 * -> 137:139
REJECT udp ----- * 0.0.0.0/0 0.0.0.0/0 137:139 -> *
MASQ all ----- * 192.168.6.20/24 0.0.0.0/0 n/a
Chain output (policy ACCEPT: 6 packets, 439 bytes):
```

Die hier aufgeführten Regeln entsprechen der Voreinstellung. Sie schützen eventuell auf der Firewall vorhandene Dienste vor dem Verbindungsaufbau aus dem Internet und vermitteln lediglich Pakete weiter, die von Adressen aus dem lokalen Netz stammen und nichts mit dem Windows-Freigabedienst (NetBIOS, SMB, CIFS, ...) zu tun haben.

Lediglich Pakete an den DNS-Server und einen (standardmäßig nicht vorhandenen) identd werden von der Firewall angenommen. Außerdem besteht prinzipiell die Gefahr, daß es einem Angreifer gelingt, Pakete zur Firewall zu schleusen, deren Absenderadressen gefälscht sind und die aus dem lokalen Netz zu stammen scheinen. Solche Pakete würden maskiert und dann weiter ins lokale Netz vermittelt. Durch das Masquerading würden sie dort als Pakete der Firewall erscheinen.

Der Aufwand, um dies zu erreichen, wäre allerdings recht hoch. Insbesondere stellt sich für den Angreifer das Problem, daß Router im Internet normalerweise keine Pakete weiterleiten, die an Adressen gerichtet sind, wie wir sie im lokalen Netz verwenden. Grundsätzlich können wir aber nicht ausschließen, daß er trotzdem einen Weg findet, die betroffenen Router zu überlisten. Damit könnte der Angreifer Pakete zwar senden, eventuelle Antworten aber nicht empfangen, so daß zwar DoS-Angriffe denkbar wären, ein Einbruch in die Rechner im lokalen Netz aber wenig praktikabel erscheint.

Als nächstes wollen wir einmal nachsehen, auf welchen Ports Server aktiviert wurden. Da auch bei fli4l das Kommando netstat fehlt, müssen wir auch hier auf die Rohdaten

des Kernels zurückgreifen. Wir finden diese in den virtuellen⁶ Dateien */proc/net/tcp* und */proc/net/udp*.

Beginnen wir mit den TCP-Ports:

```
# cat /proc/net/tcp
sl local_address rem_address  st tx_queue rx_queue tr tm->when
retrnsmt uid timeout inode
0: 00000000:1388 00000000:0000 0A 00000000:00000000 00:00000000
00000000 0 0 276
```

Hier haben wir offenkundig einen Server auf Port 5000 (hexadezimal: 1388). Dies ist der Port, den der *imond* benutzt.

Nun zu den UDP-Ports:

```
# cat /proc/net/udp
sl local_address rem_address  st tx_queue rx_queue tr tm->when
retrnsmt uid timeout inode
0: 00000000:0202 00000000:0000 07 00000000:00000000 00:00000000
00000000 0 0 244
1: 00000000:0035 00000000:0000 07 00000000:00000000 00:00000000
00000000 0 0 237
```

Port 53 (hexadezimal: 35) ist für DNS reserviert. Da wir einen DNS-Server konfiguriert haben, ist auch daran nichts ungewöhnlich. Port 514 (hexadezimal: 202) ist vergeben für das Syslog-Protokoll. Er wird wahrscheinlich vom *syslogd* verwendet.

Werfen wir nun einen Blick auf die aktiven Prozesse, so finden wir dort auch die Server zu den offenen Ports:

```
# ps
PID Uid Gid State Command
1 root root S [swapper]
2 root root S [kflushd]
3 root root S [kupdate]
4 root root S [kswapd]
5 root root S [keventd]
256 root root S sh
258 root root S /usr/sbin/pppd usepeerdns pty exec
/usr/sbin/pppoe -p
282 ens ens S /usr/local/ens/ens -T 70 -u ens -r
/usr/local/ens/ -c
288 root root S /usr/sbin/syslogd
296 root root S /usr/sbin/klogd -c 1
355 root root S /usr/local/bin/imond -log-to-syslog -port 5000
364 root root S -sh
378 root root R ps
```

Hierbei ist */usr/local/ens/ens* der von *fli4l* verwendete DNS-Server. Wie wir sehen, läuft er nicht mit Rootrechten, sondern mit den Rechten des Benutzers *ens*. Dies begrenzt den Schaden, den ein Angreifer anrichten kann, wenn er eine Sicherheitslücke

⁶ Diese Dateien sind in Wirklichkeit Bereiche im Hauptspeicher, in denen der Kernel Konfigurationseinstellungen und Angaben zum Zustand des Systems ablegt. Durch Schreiben in einige dieser Dateien kann man auch das Verhalten des Kernels beeinflussen (siehe Kapitel 8, Unterabschnitt *Konfiguration des /proc-Dateisystems*, ab Seite 166).

findet, die es ihm erlaubt, den `ens` zu veranlassen, beliebige Anweisungen auszuführen. Zusätzlich führt der `ens` auch noch einen Aufruf der Betriebssystemfunktion `chroot()` durch, die dazu führt, daß von diesem Zeitpunkt an Dateien außerhalb des Verzeichnisses `/usr/local/ens/` für ihn nicht mehr existieren.

Alle zum `ens` gehörenden Dateien finden wir unter `/usr/local/ens/`. Dort finden wir z. B. die Konfigurationsdatei `ens.conf`. Diese enthält neben der eigentlichen Datenbank des DNS-Servers auch mehrere Zeilen, die den Zugriff auf den Server regeln:

```
# cat /usr/local/ens/ens.conf
[...]
ACL dns.allow 192.168.6.
[...]
ACL dns.allow 127.0.0.1
ACL dns.deny $
```

Wie wir sehen, ist auf diese Weise zumindest sichergestellt, daß Rechner im Internet nicht die Datenbank unseres Servers auslesen.

Wollen wir überprüfen, ob unsere Vermutungen richtig sind, welche Ports von welchen Diensten verwendet werden, so können wir dies tun, indem wir uns die offenen Dateien eines Prozesses unter `/proc/<PID>/fd` ansehen. Der `syslogd` hat z. B. die PID 288:

```
# ls -l /proc/288/fd
lrwx----- 1 root  root  64 Feb 22 22:38 0 -> socket:[242]
lrwx----- 1 root  root  64 Feb 22 22:38 1 -> socket:[244]
lrwx----- 1 root  root  64 Feb 22 22:38 2 -> socket:[293]
l-wx----- 1 root  root  64 Feb 22 22:38 3 -> /dev/tty9
lrwx----- 1 root  root  64 Feb 22 22:38 4 -> socket:[297]
```

Wir stellen fest, daß der Filedescriptor 1 auf einen Socket mit der Nummer 244 zeigt. Diese Socketnummer haben wir in `/proc/net/udp` für den Port 514 gefunden. Der Port wird also wirklich vom `syslogd` verwendet.

Abschließend wollen wir noch kurz nachsehen, welche Benutzerkonten `fli4l` kennt:

```
# cat /etc/passwd
root:x:0:0:root:/:/bin/sh
fli4l:x:0:0:ftp account:/:/bin/sh
ens:x:1:100:/:/dev/null:/bin/true
nobody:x:-2:-2:/:/dev/null:/bin/true
```

Wie wir sehen, sind nur die unbedingt notwendigen Benutzer definiert. Darüber hinaus haben die beiden Konten für Systemdienste (`ens`, `nobody`) als Shell `/bin/true` eingetragen. Jeder Versuch, sich normal am System anzumelden, würde dazu führen, daß als Shell ein Programm ausgeführt wird, daß sich sofort beendet. Auch die in vielen Programmen vorgesehene Möglichkeit, direkt Befehle auszuführen, würde an diesem Umstand scheitern.

Damit ergibt sich im großen und ganzen ein positives Bild. Das System wurde mit Blick auf Sicherheitsfragen ausgelegt. Zwar bestehen noch Verbesserungsmöglichkeiten, der damit zu erzielende Sicherheitsgewinn steht aber in keinem vernünftigen Verhältnis zum Aufwand.