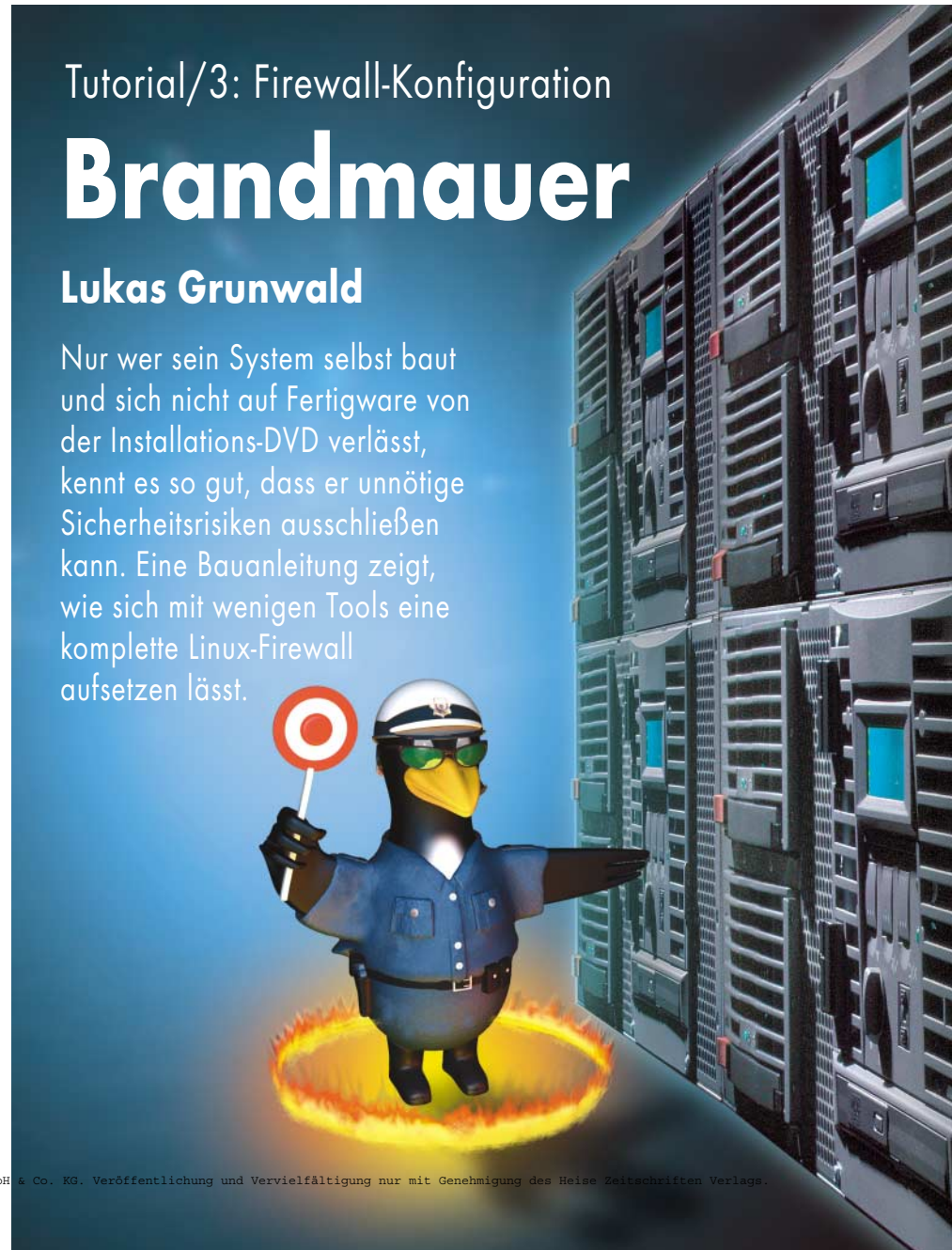
Tutorial/3: Firewall-Konfiguration

Brandmauer

Lukas Grunwald

Nur wer sein System selbst baut und sich nicht auf Fertigware von der Installations-DVD verlässt, kennt es so gut, dass er unnötige Sicherheitsrisiken ausschließen kann. Eine Bauanleitung zeigt, wie sich mit wenigen Tools eine komplette Linux-Firewall aufsetzen lässt.



Will man sich nicht darauf verlassen, dass man bei einer Installation einer fertigen Distribution alle Sicherheitslöcher schließen kann, kann man alternativ dazu ein eigenes minimales Linux-System selbst aufsetzen.

Die Beispielkonfiguration geht von folgendem Szenario aus: die Linux-Box dient als Paketfilter und Firewall, alle eingehenden Verbindungen in das Intranet sind blockiert, ausgehende FTP- und HTTP-Anfragen gehen über einen transparenten Proxy, SSL- und alle anderen ausgehenden Verbindungen sind erlaubt.

Ein Konsolenserver – über eine serielle Schnittstelle mit der Firewall verbunden – dient zur Outband-Administration (s. „Administratorzugang: gemeinsam oder getrennt“), zum Ändern des Regelsatzes und zum Weiterleiten von Log-Meldungen zu einem Syslog-Server. Die Systemzeit ist via NTP synchronisiert, die Linux-Box verfügt über keine weiteren Dienste im Userspace. Auf Benutzer, ein Init-System und Anwendungen, die den Paketfilter aufblähen, wird gezielt verzichtet. Firewall und Paketfilter sind mit *iptables* realisiert.

Ein Vorteil von *iptables* gegenüber anderen Firewalls besteht darin, dass sie keine aufwendige Umgebung erfordert: Mit einer Hand voll Komponenten lässt sich ein System bauen, das auf eine Diskette passt. Neben einem monolithischen Kernel braucht man einen Bootloader wie *syslinux* oder GRUB, will man nicht von Ersterem booten. Für schlanke Systeme stellt außerdem die Kommandosammlung *busybox* auf Grundlage der *ash* eine Benutzerumgebung mit den wichtigsten Kommandos als so genanntes Multi-Call-Binary zur Verfügung. Für das Initialisieren der Filterregeln sorgt *iptables*, für die Zeitsynchronisation der *ntp*-Dämon.

Weitere Komponenten sind nicht nötig. Auf einen Syslog-Dämon kann ebenso verzichtet werden wie auf den Kernel-Log-Dämon, da der Linux-Kernel beim Fehlen eines Syslog-Services automatisch alle Meldungen auf die Konsole schreibt, in diesem Fall über die serielle Verbindung auf dem Konsolenserver ausgibt. Damit das System auf dynamische Libraries verzichten kann, sind alle Programme statisch gelinkt. Das macht das System nicht nur schlanker und übersichtlicher, sondern verhindert auch Schwachstellen durch die *libc*.

Aufgrund der Größe lässt sich das System komplett in einer RAM-Disk betreiben. Dazu benutzt man eine *initrd* (Initial RAM Disk), die das Rootfilesystem enthält und deren Image als Datei neben dem Kernel auf einem bootfähigen Medium liegt. Bootloader, Kernel und Image der Firewall können dadurch auf einer Embedded-Hardwareplattform mit Flash-Speicher, einem bootfähigen USB-Stick, einer CD oder einer Floppy mit Schreibschutz untergebracht sein. Ebenso kann man das System via PXE (Preexecution Environment) übers Netz starten und damit ein Cold-Standby-System aufbauen, das sich ohne Softwareinstallation etwa über einen Power-Switch hochfahren lässt, wenn das Erstsystem ausfällt. In allen Fällen sind als einzige Schnittstellen für den Betrieb die serielle Verbindung zum Konsolenserver und die Netzwerk-Interfaces vonnöten.

Um ein derartiges System bauen zu können, braucht man einen separaten Linux-Rechner mit RAM-Disk-Unterstützung, GCC (Gnu Compiler Collection), Kernelquellbaum und Openwall-Patches. Dazu gesellen sich die Quellen von *busybox*, *ntp* und *iptables*, da man alle Tools als statisch gelinkte übersetzen muss. Als Beispiel soll die Firewall in einer RAM-Disk liegen und von CD starten. In diesem Fall benötigt man noch einen CD-Brenner und die *cdrtools*. Als Vorarbeit sollte man zuerst ein CD-Wurzelverzeichnis – etwa *~/CD_ROOT* – anlegen, aus dem später das CD-Image gebaut wird.

Kleine Bauarbeiten

Beim Übersetzen des Kernels sind alle Regeln des Kernelhärtens zu befolgen: In den monolithischen – also

nicht-modularen – Vanilla-Kernel mit Openwall-Patches gehören nur die Treiber, die für den Betrieb wirklich notwendig sind (s. Tutorial/1, *iX* 11/2003). Während der Kernelkonfiguration ist vor allem darauf zu achten, dass der RAM-Disk- und Initrd-Support ebenso wie *ext2* und */proc* aktiviert sind. Zusätzlich muss das Device-Filesystem *devfs* mit der Option „Automatically mount at boot" eingebunden sein, um sich das manuelle Erzeugen der Device-Dateien in */dev* zu ersparen – sonst schlägt das Booten fehl.

Soll das komplette Linux-System in einer RAM-Disk liegen, kann man auf SCSI-, IDE- respektive USB-Treiber getrost verzichten, Treiber für Festplatten oder andere Peripheriegeräte sind ebenfalls überflüssig. Lediglich die serielle Konsole sollte Unterstützung durch den Kernel finden. Außerdem ist es sinnvoll, neben den Netzwerkadaptern den „Dummy net driver support" zu aktivieren. Für die Firewall gehören *iptables* und weitere erwünschte Filter- und Routing-Funktionen sowie Protokollunterstützungen in den Kernel. *iptables* und Kernel unterstützen, wenn angewählt, sowohl IPv4- als auch IPv6- und ARP-Filtering.

Um sich nicht den Kernel und die *System.map* auf seinem System zu überschreiben, empfiehlt es sich, im Haupt-*Makefile* vorübergehend den Installationspfad auf *~/CD_ROOT* zu setzen: *export INSTALL_PATH=~/CD_ROOT*, danach kann man ruhigen Gewissens den Befehl *make dep && make && make install* eingeben.

Kofferpacken

Für das Erzeugen der Benutzerumgebung ist zuerst das Dateisystem anzulegen, für das 4 MByte ausreichen sollten. Das geschieht mit dem Befehl *mke2fs –m0 /dev/ram0 –b 1024 4096*. Die Option *–m0* sorgt dafür, dass *mke2fs* keine Blöcke für den Superuser reserviert, während *–b 1024* die verwendete Blockgröße und *4096* die Größe des Dateisystems in KByte angibt. Danach lässt es sich mit den Kommandos *mkdir /ramdisk* und *mount –t ext2 /dev/ram0 /ramdisk* mounten.

Da eine RAM-Disk immer die Gefahr des Datenverlustes durch einem Rechnerabsturz oder unbeabsichtigten Neustart birgt, kann man alternativ mit *dd if=/dev/zero of=~/CD_ROOT/initrd bs=1024 count=4096* erst die – leere – Image-Datei erstellen, diese mit *mke2fs –m0 –b 1024 ~/CD_ROOT/initrd* formatieren und mit *mount –o loop ~/CD_ROOT/initrd /ramdisk* als Loop-Device ins Dateisystem einhängen sowie bearbeiten. Danach geht es ans Übersetzen der benötigten Utilities.

Die Busybox-Quellen bietet seit dem Versionssprung von 0.60.x auf 1.00-prex eine umfangreiche und vor allem benutzerfreundliche Konfiguration. Während man bei den Versionen 0.60.x die Header-Datei *Config.h* im Wurzelverzeichnis des Quellbaums editieren oder nach einem Aufruf von *make config* die Datei *applet_source_list* entrümpeln musste, liefert die momentan aktuelle Version 1.00-pre3 eine übersichtliche *.config*-Datei und ein dialogbasiertes Konfigurationsskript mittels *make menuconfig* – ähnlich wie beim Linux-Kernel – mit (s. Listing 1).

Da alle erwünschten Kommandos, Optionen und Funktionen in das *busybox*-Binary beim Kompilieren und Linken fest eingebunden werden und man bei falscher Konfiguration die gesamte Arbeit wiederholen muss, ist hier wie beim Kernelbauen ein aufmerksames Durcharbeiten der Konfiguration angebracht. Im Fall der Firewall muss die Option „Build BusyBox as a static binary" eingeschaltet sein, dafür sollten sämtliche Kommandos und Funktionen in den Sektionen „Archival Utilities", „Login/Password Management Utilities", „Miscellaneous Utilities" und „Linux Module Utilities" deaktiviert sein.

Dasselbe gilt für Tools zur Systemmanipulation wie *chgrp*, *chmod*, *chown*, *chroot* und für Datei- und Filesystem-Utilities wie *cp cut*, *dd*, *ln, mv*, *rm*, *mkdir*, *mknod*, *sort*, *touch*, *uniq*, *which*, *sed*, *find*, *grep*. Während man *umount* ebenfalls als überflüssig einstufen kann, muss *mount* für das Einhängen des */proc*-Dateisystem aktiviert sein.

Außerdem benötigt das System die Kommandos *md5sum*, *mkfifo*, *watch*, *hostid* und die *init*-Option „Support running init from within an initrd".

Bei den „Networking Utilities" ist darauf zu achten, dass *arping*, *ip* und seine erweiterten Funktionen sowie *ipaddr*, *iplink*, *iproute* und *iptunnel* eingebunden sind. Mit den „Logging Utilities" *syslogd*, *klogd* und *logger* lässt sich ein komplettes Syslogging einrichten, das aber auf der Beispiel-Firewall deaktiviert ist, da alle Kernelmeldungen über die Konsole laufen und über den Konsolenserver ihren Weg zum Syslog-Server finden. Wenn man sich hier für ein Inband-Management entscheidet, ist es sinnvoll, die Tools ebenfalls mit einzubinden.

Nach dem Übersetzen mit *make* kann man durch Angabe des Zielverzeichnisses */ramdisk* die Busybox mit *make PREFIX=/ramdisk install* direkt auf die RAM-Disk installieren. Der Befehl erzeugt die Verzeichnisse */ramdisk/bin*, */ramdisk/sbin* und */ramdisk/usr/bin* sowie für sämtliche verfügbaren Kommandos Links auf das *busybox*-Binary in */ramdisk/bin*.

Dämonenzucht

Mit einem Wechsel in das *ntp*-Quellverzeichnis beginnt das Übersetzen des *ntpd*. Um den NTP-Dämon statisch zu linken, muss man dem Aufruf *./configure* entsprechende *LDFLAGS* und *CFLAGS* übergeben: *CFLAGS='–O' LDFLAGS='––static –s' ./configure ––disable–all–clocks ––disable–debugging* Die Option *–s* erspart das nachträgliche Entfernen der Debug-Informationen und die Symboltabelle mit *strip ntpd/ntpd*, was die Größe des Binary drastisch reduziert. Die Optionen *––disable–all–clocks* und *––disable–debugging* entfernen Uhrentreiber und

iX-TRACT

- Für eine Minimalinstallation einer Linux-Firewall benötigt man neben Kernel und *busybox* nur *iptables* und *ntpd*.
- Auf jeden Fall sollte man die zu verwendenden Binaries selbst übersetzen, um sie von Grund auf anzupassen.
- Mit einem eigenen Startskript lässt sich ein minimiertes Init-System aufsetzen, das wenig Spielraum für Manipulationen lässt.

Tutorial-Übersicht

Teil 1: Härten und Abspecken eines sicherheitsrelevanten Linux-Systems am Beispiel eines minimalen statischen Webservers

Teil 2: Erstellen von Ereignismitschnitten und Auswertung von Log-Dateien

Teil 3: Konfigurieren und Härten einer Firewall am Beispiel von Linux; Planung, Dokumentation und Aufbau von Audit-Kreisläufen

Listing 1: /usr/src/packages/busybox-1.00-pre3/.config

```
# Automatically generated make config
#
HAVE_DOT_CONFIG=y                       # .config
#
# General Configuration
#
CONFIG_FEATURE_BUFFERS_GO_ON_STACK=y    # Buffer allocation policy (Allocate on the Stack)
CONFIG_FEATURE_DEVFS=y                  # Support for devfs
CONFIG_FEATURE_DEVPTS=y                 # Use the devpts filesystem for Unix98 PTYs
#
### Build Options
#
CONFIG_STATIC=y                         # Build BusyBox as a static binary (no shared libs)
#
### Archival Utilities                  # disable all
#
# CONFIG_GUNZIP is not set              # disable gunzip
# CONFIG_GZIP is not set                # disable gzip
# CONFIG_TAR is not set                 # disable tar
#
### Coreutils
#
# CONFIG_BASENAME is not set            # disable basename
CONFIG_CAT=y                            # cat
# CONFIG_CHGRP is not set               # disable chgrp
# CONFIG_CHMOD is not set               # disable chmod
# CONFIG_CHOWN is not set               # disable chown
# CONFIG_CHROOT is not set              # disable chroot
# CONFIG_CP is not set                  # disable cp
# CONFIG_CUT is not set                 # disable cut
CONFIG_DATE=y                           # date (forced enabled for use with watch)
CONFIG_FEATURE_DATE_ISOFMT=y            # date: enable ISO date format output (-I) of date
# CONFIG_DD is not set                  # disable dd
# CONFIG_DF is not set                  # disable df
# CONFIG_DIRNAME is not set             # disable dirname
CONFIG_DU=y                             # du
CONFIG_FEATURE_DU_DEFALT_BLOCKSIZE_1K=y # du: use a default blocksize of 1024 bytes (1K)
CONFIG_ECHO=y                           # echo
CONFIG_FEATURE_FANCY_ECHO=y             # echo: enable echo options (-n and -e)
CONFIG_ENV=y                            # env
CONFIG_FALSE=y                          # false (forced enabled for use with shell)
CONFIG_HEAD=y                           # head
CONFIG_HOSTID=y                         # enable hostid
CONFIG_ID=y                             # id
# CONFIG_LN is not set                  # disable ln
CONFIG_LS=y                             # ls
CONFIG_FEATURE_LS_FILETYPES=y           # ls: filetyping options (-p and -F)
CONFIG_FEATURE_LS_FOLLOWLINKS=y         # ls: symlinks dereferencing (-L)
CONFIG_FEATURE_LS_SORTFILES=y           # ls: sort the file names
CONFIG_FEATURE_LS_TIMESTAMPS=y          # ls: show file timestamps
CONFIG_FEATURE_LS_USERNAME=y            # ls: show username/groupnames
CONFIG_FEATURE_LS_COLOR=y               # ls: use color to identify file types
CONFIG_MD5SUM=y                         # enable md5sum
# CONFIG_MKDIR is not set               # disable mkdir
CONFIG_MKFIFO=y                         # enable mkfifo
# CONFIG_MKNOD is not set               # disable mknod
# CONFIG_MV is not set                  # disable mv
CONFIG_PWD=y                            # pwd
# CONFIG_RM is not set                  # disable rm
# CONFIG_RMDIR is not set               # disable rmdir
CONFIG_SHA1SUM=y                        # sha1sum
CONFIG_FEATURE_SHA1SUM_CHECK=y          # sha1sum: enable -c and -w options
CONFIG_SLEEP=y                          # sleep (single integer arg with no suffix)
# CONFIG_SORT is not set                # disable sort
CONFIG_SYNC=y                           # sync
CONFIG_TAIL=y                           # tail
CONFIG_TEST=y                           # test (forced enabled for use with shell)
# CONFIG_TOUCH is not set               # disable touch
CONFIG_TRUE=y                           # true (forced enabled for use with shell)
# CONFIG_TTY is not set                 # disable tty
CONFIG_UNAME=y                          # uname
# CONFIG_UNIQ is not set                # disable uniq
CONFIG_WATCH=y                          # enable watch
# CONFIG_WHOAMI is not set              # disable whoami
# CONFIG_YES is not set                 # disable yes
#
## Common options for ls and more
#
CONFIG_FEATURE_AUTOWIDTH=y              # calculate terminal & column widths
#
## Common options for df, du, ls
#
CONFIG_FEATURE_HUMAN_READABLE=y         # support for human readable output (13k, 23M, 235G)
#
### Console Utilities
#
CONFIG_CLEAR=y                          # clear
CONFIG_RESET=y                          # reset
CONFIG_OPENVT=y                         # openvt
#
### Debian Utilities                    # disable all
#
# CONFIG_WHICH is not set               # disable which
#
### Editors                             # disable all
#
# CONFIG_SED is not set                 # disable sed
#
### Finding Utilities                   # disable all
#
# CONFIG_FIND is not set                # disable find
# CONFIG_GREP is not set                # disable grep
#
### Init Utilities
#
CONFIG_INIT=y                           # init
CONFIG_FEATURE_USE_INITTAB=y            # support reading an inittab file?
CONFIG_FEATURE_INITRD=y                 # enable "Support running init from within an initrd"
CONFIG_HALT=y                           # halt
CONFIG_POWEROFF=y                       # poweroff
CONFIG_REBOOT=y                         # reboot
CONFIG_MESG=y                           # mesg
#
### Login/Password Management Utilities # disable all
#
#
### Miscellaneous Utilities             # disable all
#
#
### Linux Module Utilities              # disable all
#
#
### Networking Utilities
#
CONFIG_ARPING=y                         # enable arping
CONFIG_HOSTNAME=y                       # hostname
# CONFIG_IFCONFIG is not set            # disable ifconfig
CONFIG_IP=y                             # enable ip
CONFIG_FEATURE_IP_ADDRESS=y             # ip: address (forced enabled for ipaddr)
CONFIG_FEATURE_IP_LINK=y                # ip: link (forced enabled for iplink)
CONFIG_FEATURE_IP_ROUTE=y               # ip: route (forced enabled for iproute)
CONFIG_FEATURE_IP_TUNNEL=y              # ip: tunnel (forced enabled for iptunnel)
CONFIG_IPADDR=y                         # enable ipaddr
CONFIG_IPLINK=y                         # enable iplink
CONFIG_IPROUTE=y                        # enable iproute
CONFIG_IPTUNNEL=y                       # enable iptunnel
CONFIG_NSLOOKUP=y                       # nslookup
CONFIG_PING=y                           # ping
CONFIG_FEATURE_FANCY_PING=y             # ping: fancy ping output
CONFIG_ROUTE=y                          # route
#
### Process Utilities
#
CONFIG_FREE=y                           # free
CONFIG_KILL=y                           # kill
CONFIG_KILLALL=y                        # killall
CONFIG_PS=y                             # ps
CONFIG_UPTIME=y                         # uptime
#
### Another Bourne-like Shell
#
CONFIG_FEATURE_SH_IS_ASH=y              # Choose your default shell (ash)
#
## Ash Shell Options
#
CONFIG_ASH_JOB_CONTROL=y                # Job control
CONFIG_ASH_ALIAS=y                      # alias support
CONFIG_ASH_MATH_SUPPORT=y               # Posix math support
# CONFIG_ASH_MAIL is not set            # disable "Check for new mail on interactive shells"
CONFIG_ASH_OPTIMIZE_FOR_SIZE=y          # optimize for size instead of speed
CONFIG_HUSH=y                           # enable hush
CONFIG_LASH=y                           # enable lash
CONFIG_MSH=y                            # enable msh
#
## Bourne Shell Options
#
CONFIG_FEATURE_COMMAND_EDITING=y        # command line editing
CONFIG_FEATURE_COMMAND_TAB_COMPLETION=y # tab completion
CONFIG_FEATURE_COMMAND_HISTORY=15       # history size: 15 records
CONFIG_FEATURE_SH_STANDALONE_SHELL=y    # enable standalone shell
CONFIG_FEATURE_SH_FANCY_PROMPT=y        # enable fancy shell prompts
CONFIG_FEATURE_SH_EXTRA_QUIET=y         # enable hide message on interactive shell startup
#
### System Logging Utilities
#
# CONFIG_SYSLOGD is not set             # disable syslogd
# CONFIG_KLOGD is not set               # disable klogd
# CONFIG_LOGGER is not set              # disable logger
#
### Linux System Utilities
#
CONFIG_DMESG=y                          # dmesg
CONFIG_MORE=y                           # more
# CONFIG_FEATURE_USE_TERMIOS is not set # disable "Use termios to manipulate the screen"
CONFIG_MOUNT=y                          # mount
# CONFIG_UMOUNT is not set              # disable umount
```

weitere Debugging-Information aus dem Binary. Wer eine Hardwareuhr oder einen DCF77-Empfänger an der Firewall anschließen will, verzichtet besser auf die Option *––disable–all–clocks*. Nach dem Übersetzen sollte das Kommando *file ntpd/ntpd* das ELF-Binary dann als *statically linked, stripped* ausweisen.

Ähnliches gilt für das Tool *iptables*, auch das muss statisch gelinkt sein. Dazu genügt beim Übersetzen die Option *NO_SHARED_LIBS=1*. Da der Präprozessor beim Übersetzen auf das *include*-Verzeichnis des Kernelquellbaums zugreifen muss, sollte der unter */usr/src/linux* liegen oder ein entsprechender Link existieren. Andernfalls kann man dem *gcc* mit *KERNEL_DIR=<pfad>* den richtigen Weg weisen: *make NO_SHARED_LIBS=1 KERNEL_DIR=/usr/src/linux-2.4.23* Danach empfiehlt es sich, mit *strip iptables* die Größe des Binary zu reduzieren. Das darauffolgende Kommando *file iptables* muss ebenfalls *statically linked, stripped* ergeben.

Das Kopieren der Binaries *iptables* und *ntpd* nach */ramdisk/bin* schließt die Übersetzungsarbeiten ab. Es empfiehlt sich unter Umständen, eine Sicherungskopie oder einen Hardlink beider Binaries unter */ramdisk/usr/local/bin* abzulegen, falls man mit der *busybox*-Konfiguration experimentieren und die alten *busybox*-Verzeichnisse auf der RAM-Disk löschen will.

Nun sind die Verzeichnisse */ramdisk/dev*, */ramdisk/proc*, */ramdisk/tmp*, */ramdisk/etc* und */ramdisk/etc/init.d* anzulegen. Außerdem benötigt das System zum Starten die Dateien *inittab*, *rcS*, *fstab* und *ntp.conf*, eine *profile*-Datei ist optional.

Busybox enthält ein einfaches Init-System, das eine vereinfachte Syntax benutzt. Das *examples*-Verzeichnis des *busybox*-Quellbaums enthält eine beispielhafte *inittab*-Datei, die man am besten nach */ramdisk/etc* kopiert und editiert: Neben allen zusätzlichen TTYs sollten die *umount*- und *swapoff*-Einträge entfernt werden (s. Listing 2). Außerdem befinden sich unter *examples/bootfloppy/etc* weitere Beispieldateien.

Im Verzeichnis */ramdisk/etc/init.d* ist nun eine Datei *rcS* anzulegen, die das Startskript der Linux-Box enthält, angestoßen von der *inittab* (s. Listing 3): Dem Aufruf von *mount –a* folgt die Integritätsprüfung der Binaries *busybox*, *ntpd*, *iptables* und des Startskripts selbst. Nach dem Initialisieren der Netzwerk-Interfaces startet der *ntpd*, ihm folgt das Erstellen der *iptables*-Regeln. Ihre Syntax folgt in der Regel dem Muster *iptables –option chain rule-specification target* (s. Listing 3). Aus Gründen der Übersichtlichkeit beschränkt sich das Listing auf IPv4; Filterregeln für ARP und IPv6 lassen sich analog erstellen.

Regeln erstellen mit Plan

Damit man nicht wahllos Regeln in der Firewall einträgt und ändert, ist der Regelsatz zuvor durch eine dokumentierte Policy niederzuschreiben. Sie definiert, welche Ports von welchen Servern oder Clients unter welchen Umständen zu erreichen sein dürfen, welche Ports in die DMZ umgeleitet werden et cetera.

Erst wenn diese Dokumentation verfasst ist, kann man das Startskript *rcS* an die Regeln anpassen. Nachdem die Firewall in Betrieb genommen wird, ist darauf zu achten, dass die aufgestellten Regeln für den Paketfilter tatsächlich aktiv sind.

Hält man sich an die Vorgehensweise von Definition auf Papier, Umsetzung und Kontrolle mit *nmap*, wie sie im Audit-Kreislauf abgebildet ist (s. Tutorial/1, *iX* 11/2003, S. 46), ist jede Änderung auch für Drtte nachvollziehbar.

Die Datei */etc/fstab* – ebenfalls im *etc*-Verzeichnis – enthält lediglich die Zeile *proc /proc proc defaults 0 0*, durch die *mount –a* das */proc*-Dateisystem einbindet.

Damit *ntpd* seine Arbeit aufnehmen kann, möchte er im *etc*-Verzeichnis eine Datei namens *ntp.conf* vorfinden. Ihre Syntax hat der zweite Teil des Tutorials vorgestellt (s. *iX* 12/2003, S. 136). In der *ntp.conf* sollten sich neben einem Eintrag „*restrict default ignore*“ nur Einträge mit abzufragenden Zeitservern befinden.

Wer eine Warnung am Login platzieren möchte, kann das durch Anlegen einer *profile*-Datei in */ramdisk/etc* mit dem Eintrag *echo; echo „WARNING: This machine is permanently monitored. Login only for administrators !!“; echo; uname* erreichen.

listing 2: /etc/inittab

```
# /etc/inittab init(8) configuration for BusyBox
#
# Format for each entry: <id>:<runlevels>:<action>:<process>
::sysinit:/etc/init.d/rcS

# Start an "askfirst" shell on the console (whatever that may be)
::askfirst:-/bin/sh

# Stuff to do when restarting the init process
::restart:/sbin/init

# Stuff to do before rebooting
::ctrlaltdel:/sbin/reboot
```

Nachdem sich alle benötigten Dateien auf ihrem Platz befinden, kann man das RAM-Disk-Image mit *umount /ramdisk* und *dd if=/dev/ram0 of=~/CD_ROOT/initrd bs=1024 count=4096* erstellen und mit *gzip ~/CD_ROOT/initrd* komprimieren. Wer das Image bereits erstellt und als Loop-Device eingehängt hat, kommt mit *umount /ramdisk* und *gzip ~/CD_ROOT/initrd* aus.

Jetzt benötigt man nur noch einen Bootloader, der zusammen mit Kernel und *initrd.gz* das Bootmedium füllt, in diesem Fall eine CD. Wer mit *grub* oder *lilo* arbeitet, findet in den Paketen eine ausführliche Beschreibung der Installation auf das Bootmedium. Dabei ist zu bedenken, dass beide Bootloader für solch ein Minimalsystem auf Diskette, CD oder Flash-Speicher überdimensioniert sind.

Wer dennoch nicht auf seinen Lieblings-Bootloader verzichten will, sollte – im Falle des *grub* – die Zeilen *serial ––unit=0 ––speed=9600* und *terminal ––timeout=15 console serial* in der *menu.lst* respektive die Zeile *append=“console=/dev/ttyS0,9600* sowie *seriell=0,9600n8* in der *lilo.conf* nicht vergessen.

Wesentlich einfacher als mit *lilo* oder *grub* lässt sich eine bootfähige CD nach „El Torito“ mit dem Paket *syslinux* erstellen. Den dafür benötigten Bootsektor liefert *syslinux* unter dem Namen *isolinux.bin* mit, den man nach *~/CD_ROOT* kopiert. Erstellen muss man nur noch die ASCII-Datei *~/CD_ROOT/isolinux.cfg* für die Bootoptionen. Sie folgt der Syntax der sonst üblichen *syslinux.cfg*. Wichtig sind auch hier die Einträge *SERIAL 0 9600* und *APPEND console=ttyS0* für die serielle Konsole (s. Listing 4).

Der Befehl *mkisofs –o fw.img –b isolinux.bin –c boot.cat –no–emul–*

WICHTIGE URLS

www.busybox.net	Busybox-Homepage
www.netfilter.org	Netfilter/Iptables-Projekt
www.ntp.org	NTP-Projekt
syslinux.zytor.com	Syslinux-Homepage
www.kernel.org	Kernel-Homepage
www.openwall.com/linux	Openwall-Patches

listing 3:/etc/init.d/rcS

```
#! /bin/sh
echo "STATUS BOOTING
mount -a
echo CHECKING-Integrity
md5sum /bin/busybox
md5sum /bin/ntpd
md5sum /bin/iptables
md5sum /etc/init.d/rcS

echo "STARTING  NETWORKING"
# Loopback Interface
ip link set lo up
ip addr add 127.0.0.1/8 dev lo
ip link show dev lo

# Internet-Interface
ip link set eth0 up
ip addr add 191.168.181.37/25 dev eth0
ip link show dev eth0

# DMZ-Interface
ip link set eth1 up
ip addr add 192.168.10.1/24 dev eth1
ip link show dev eth1

# IntraNET-Interface
ip link set eth2 up
ip addr add 192.168.20.1/24 dev eth2
ip link show dev eth2

echo "STARTING NTP SERVICES"
/bin/ntpd

echo "SETTING IPTABLES RULES"
/bin/iptables -F
/bin/iptables -X

/bin/iptables -N LOG_DROP
/bin/iptables -A LOG_DROP -j LOG  --log-prefix '[IPTABLES DROP] : '
/bin/iptables -A LOG_DROP -j DROP

/bin/iptables -N LOG_ACCEPT
/bin/iptables -A LOG_ACCEPT -j LOG --log-prefix '[IPTABLES ACCEPT] : '
/bin/iptables -A LOG_ACCEPT -j ACCEPT

/bin/iptables -P INPUT DROP
/bin/iptables -P OUTPUT DROP
/bin/iptables -P FORWARD DROP

/bin/iptables -A INPUT  -i lo -j ACCEPT
/bin/iptables -A OUTPUT -o lo -j ACCEPT

# Blocking some networks global
/bin/iptables -A INPUT -s 213.20.0.0/16 -d 0/0 -j DROP
/bin/iptables -A FORWARD -s 213.20.0.0/16 -d 0/0 -j DROP
/bin/iptables -A INPUT -s 212.24.128.0/19 -d 0/0 -j DROP
/bin/iptables -A FORWARD -s 212.24.128.0/19 -d 0/0 -j DROP

/bin/iptables -A INPUT -i eth0 -s 10.0.0.0/8 -j DROP
/bin/iptables -A INPUT -i eth0 -s 172.16.0.0/12 -j DROP
/bin/iptables -A INPUT -i eth0 -s 192.168.0.0/16 -j DROP

# Block some XMAS and other TCP Scan mechanism
/bin/iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP
/bin/iptables -A FORWARD -p tcp --tcp-flags ALL ALL -j DROP
# Block Null packets
/bin/iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
/bin/iptables -A FORWARD -p tcp --tcp-flags ALL NONE -j DROP

# Transparent Squid Proxy
# proxy ( run on the firewall) have illimit access to internet (www)
/bin/iptables -A OUTPUT -o eth2 -m state --state NEW,ESTABLISHED -p tcp --dport 80 -j ACCEPT
/bin/iptables -A INPUT -i eth2 -m state --state ESTABLISHED -p tcp --sport 80 -j ACCEPT

# Redirect all HTTP-Requests to the Transparent SQUID Proxy in the DMZ 192.168.10.2
/bin/iptables -t nat -A PREROUTING -i eth2 -s ! 192.168.10.2 -p tcp --dport 80 -j DNAT --to-destination
192.168.10.2:3128

# IP vom Squid-Proxy extern auf 42.42.42.42 umbiegen !
/bin/iptables -t nat -A POSTROUTING -o eth2 -s 192.168.10.0/24 -d 192.168.10.2 -j SNAT --to 42.42.42.42
/bin/iptables -A FORWARD -i eth1 -s 192.168.10.2 -d 192.168.20.0/24 -p tcp --sport 3128 -j ACCEPT
/bin/iptables -A FORWARD -i eth2 -s 192.168.10.0/24 -d 192.168.10.2 -p tcp --dport 3128 -j ACCEPT

# Allow my Name-Server from my ISP 193.217.65.2
/bin/iptables -A FORWARD -i eth0 -p udp --dport 53 -d 193.217.65.2 -j ACCEPT
/bin/iptables -A FORWARD -i eth0 -p udp --sport 53 -s 193.217.65.2 -j ACCEPT
/bin/iptables -A FORWARD -i eth1 -p udp --dport 53 -d 192.168.10.2 -j ACCEPT
/bin/iptables -A FORWARD -i eth1 -p udp --sport 53 -s 192.168.10.2 -j ACCEPT

# Allow some ICMP Traffic
/bin/iptables -A FORWARD -i eth0 -p icmp -s 0/0 --icmp destination-unreachable -d 0/0 -j ACCEPT
/bin/iptables -A FORWARD -i eth0 -p icmp -s 0/0 --icmp time-exceeded -d 0/0 -j ACCEPT
/bin/iptables -A FORWARD -i eth0 -p icmp -s 0/0 --icmp echo-reply -d 0/0 -j ACCEPT

# Allow my Internet-NTP and external NTP Services
# ntp1.pdb.de Timeserver
/bin/iptables -A INPUT -i eth0 -p udp -s 192.53.103.103 -d 191.168.181.37 -j ACCEPT
# ntp2.pdb.de Timeserver
/bin/iptables -A INPUT -i eth0 -p udp -s 192.53.103.104 -d 191.168.181.37 -j ACCEPT
# ntp2.usno.navy.mil Timeserver
/bin/iptables -A INPUT -i eth0 -p udp -s 192.5.41.209 -d 191.168.181.37 -j ACCEPT
# ntp1.usno.navy.mil  Timeserver
/bin/iptables -A INPUT -i eth0 -p udp -s 192.5.41.41 -d 191.168.181.37 -j ACCEPT

# Allow Acces to the Webserver in the DMZ
/bin/iptables -t nat -A PREROUTING -d 62.251.190.244 -p tcp --dport 80 -j DNAT --to-destination 192.168.2.2:80

# LAN ---> internet
/bin/iptables -A FORWARD -i eth2 -o eth0 -j ACCEPT
/bin/iptables -A FORWARD -o eth2 -i eth0 -j ACCEPT

# LAN  ----> web server in DMZ
/bin/iptables -A FORWARD -i eth2 -o eth1 -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
/bin/iptables -A FORWARD -i eth1 -o eth2 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT

# internet ----> web server in DMZ
/bin/iptables -A FORWARD -i eth0 -o eth1 -p tcp --destination-port 80 -m state --state NEW,ESTABLISHED -j ACCEPT
/bin/iptables -A FORWARD -o eth0 -i eth1 -p tcp --source-port 80 -m state --state ESTABLISHED -j ACCEPT

# Hide the private lan with a unique IP
/bin/iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 42.42.42.42

#  Drop all packet reaching here ...
/bin/iptables -A FORWARD -j LOG_DROP -m limit --limit 100/minute
/bin/iptables -A INPUT -j LOG_DROP -m limit --limit 100/minute
/bin/iptables -A OUTPUT -j LOG_DROP -m limit --limit 100/minute

/bin/iptables -L -n -v
echo1> /proc/sys/net/ipv4/ip-forward
```

boot –boot–load–size 4 –boot–info–table ~/CD_ROOT erstellt das CD-Image *fw.img*, wobei die Option *–b* für das Kopieren des Bootsektors von *isolinux.bin* sorgt, während die Option *–c* die Katalogdatei *boot.cat* anlegt. Bei den Argumenten beider Optionen erwartet *mkisofs* relative Pfade zu *CD_ROOT*. Sind die Variablen *CDR_DEVICE* und *CDR_SPEED* in der Datei */etc/default/cdrecord* an das System angepasst, genügt das Kommando *cdrecord –v fw.img*, um die CD zu brennen.

Für das Outband-Logging der Firewall benötigt man einen *syslog*-Dämon auf dem Konsolenserver, der vom seriellen Port */dev/ttySx*, an dem die Firewall hängt und auf den sie die Meldungen ausgibt, ins Syslog-System schreiben kann. Einen solchen bringt beispielsweise der Konsolenserver Cyclades Alterpath ACS mit [1].

Listing 4: /boot/isolinux.cfg

```
# isolinux (bootable CD):
SERIAL 0                # benutzter serieller Port, ggf. <Port> <Baudrate>
TIMEOUT 0               # kein Bootprompt
KBDMAP keymap
LABEL linux             # Menüeintrag
  KERNEL vmlinuz        # Optionen zum Menüeintrag
  APPEND load_ramdisk=1 initrd=initrd.gz root=/dev/ram0 boot=/dev/cdrom console=ttyS0

# syslinux (Floppy):
# APPEND load_ramdisk=1 initrd=initrd.gz root=/dev/ram0 boot=/dev/fd0 console=ttyS0

# pxelinux only (PXE-Boot)
# IPAPPEND ip=<client-ip>:<boot-server-ip>:<gateway-ip>:<netmask>
```

Bootet man die CD auf der Firewall-Box, führt *init* nur das Skript */etc/init.d/rcS* aus. Auf der Konsole sind die MD5-Checksummen der Binaries abzulesen, ebenso wie die *iptables*-Regeln. Die Kommandos *ps aux* und *iptables –l* geben die Prozesstabelle und die *iptables*-Regeln aus. Alle Ausgaben gehen direkt an den Syslog-Server.

Da die Firewall über eine serielle Verbindung an ein Managementsystem angeschlossen ist, kann man

so eine Manipulation der Binaries oder Zugriffversuche erkennen. Einzig über den *ntpd* dürfte eine Verbindung zum Rest der Welt bestehen.

Mit angezogener Handbremse

Wer das Sicherheitsniveau weiter erhöhen möchte, kann den Userspace mit dem Befehl *halt* anhalten. Dann läuft nur noch der Kernel mit seinen Treibern, Routing- und Filterfunktionen im Arbeitsspeicher, alle Prozesse im Userspace sind dagegen beendet, der Kernel ist komplett gegen Sicherheitsfehler im Userspace geschützt.

Will man das *halt*-Kommando ans Ende etwa der *profile* anfügen, sollte man zumindest mit *echo „Halt system? press ENTER or Ctr+C“; read dummy; halt* oder *echo „Halt system? press Ctr+C to abort“; for i in 5 4 3 2 1; do echo „$i seconds to system halt“; sleep 1; done; halt* die Option einbauen, den System-Shutdown zu unterbinden, da man sonst jeglichen Zugang zum System verbaut. Der erste Befehl erfordert eine Eingabe des Administrators, um entweder den Shutdown auszulösen oder Zugang zur Shell zu bekommen, der zweite lässt dem Admin 5 Sekunden Zeit, den *halt*-Befehl abzubrechen.

Eine solche Konfiguration ist zuvor gut abzuwägen. Zum einen muss man dann auf den NTP-Dämon auf der Firewall verzichten, kann den Zeitstempel des NTP-Dämon aber im Outband-Management hinzufügen. Zum Zweiten sind etwa spezielle Informationen aus dem Kernel über das */proc*-Filesystem nicht mehr zugänglich. Zwar werden die Syslog-Meldungen weiter auf dem Konsolenserver gesammelt, allerdings ist der Regelsatz nicht mehr veränderbar. Dafür ist der Regelsatz weitestgehend gegen Manipulationen gesichert und das Risiko auf eine Softwaresicherheitslücke (Vulnerability) des Linux-Kernels reduziert. (sun)

LUKAS GRUNWALD

arbeitet als Consultant bei der DN Systems GmbH in Hildesheim und ist in diverse freie Softwareprojekte involviert.

Literatur

[1] Dirk Wetter; Terminalserver; Powered by Penguins; Cyclades Alterpath ACS-16; *iX* 4/2003, S. 58

Administratorzugang: gemeinsam oder getrennt

Es ist keineswegs selbstverständlich, dass der Administrator die gleichen Netzwerkwege benutzt wie Mitarbeiter oder Kunden. Bei der so genannten Inband-Administration sind Verwaltungsdienste wie die Secure Shell an die gleiche Internetadresse gebunden wie die benutzerbezogenen à la SMTP oder HTTP (s. Abb. oben). Das ist zwar leicht zu implementieren, doch kaum gegen unbefugte Zugriffe abgesichert. Hat ein Unbefugter die Firewall überwunden, kann er auch auf die Administrationssoftware zugreifen.

Die Outband-Administration geht von einem physisch separaten Administrationsnetzwerk aus. Der Administrator verfügt über ein eigenes Netz und einen Zugang zu den Servern zum Beispiel über einen getrennten PC an seinem Arbeitsplatz. Der PC hat keine permanente Anbindung an das Firmennetz. Damit ist sichergestellt, dass keine anderen Mitarbeiter auf den PC zugreifen und keine Viren den Computer befallen können (s. Abb. unten).

Beim Outband-Management kann man die Sicherheit weiter erhöhen, wenn man die Router oder Firewall-Systeme über einen seriellen Port an einen Konsolenserver anbindet. Auf diese Weise sind keine IP-Verbindungen für die Administration nötig, dennoch ist ein direkter Konsolenzugriff gegeben. Diese Variante bietet dem Administrator zudem die Möglichkeit, von seinem Arbeitsplatz aus das IPL (Initial Programm Load) durchzuführen, also zu booten, was via Netzwerk nicht möglich ist.