Anleitung zum Absichern von Debian

Javier Fernández-Sanguino Peña <jfs@debian.org>
'Autoren' auf dieser Seite

Version: 3.10, Fri, 12 Jan 2007 15:36:37 +0100

Zusammenfassung

Dieses Dokument handelt von der Sicherheit im Debian-Projekt und im Betriebssystem Debian. Es beginnt mit dem Prozess, eine Standardinstallation der Debian GNU/Linux-Distribution abzusichern und abzuhärten. Es deckt die gewöhnliche Arbeit ab, eine sichere Netzwerkumgebung mit Debian GNU/Linux zu schaffen, und liefert zusätzliche Informationen über die verfügbaren Sicherheitswerkzeuge. Es befasst sich auch damit, wie die Sicherheit in Debian vom Sicherheits- und Auditteam gewährleistet wird.

Copyright-Hinweis

Copyright © 2002, 2003, 2004, 2005, 2006 Javier Fernández-Sanguino Peña

Copyright © 2001 Alexander Reelsen, Javier Fernández-Sanguino Peña

Copyright © 2000 Alexander Reelsen

An manchen Abschnitten besteht ein Copyright © der jeweiligen Autoren. Genaueres erfahren Sie unter 'Danksagungen' auf Seite 25.

Es ist erlaubt, dieses Dokument unter den Bedingungen der GNU General Public License, Version 2 (http://www.gnu.org/copyleft/gpl.html) oder jeder späteren Version, die von der Free Software Foundation veröffentlicht wird, zu kopieren, zu verbreiten und/oder zu verändern. Es wird in der Hoffnung veröffentlicht, dass es sich als nützlich erweisen könnte, aber OHNE JEDE GEWÄHRLEISTUNG.

Es ist erlaubt, unveränderte Kopien dieses Dokuments zu erstellen und zu vertreiben, vorausgesetzt der Copyright-Hinweis und diese Genehmigung bleiben auf allen Kopien erhalten.

Es ist erlaubt, veränderte Kopien dieses Dokuments unter den Voraussetzungen für unverändertes Kopieren, zu erstellen und zu vertreiben, sofern die gesamte resultierende Arbeit unter den Bedingungen einer Genehmigung identisch zu dieser, vertrieben wird.

Es ist erlaubt, Übersetzungen dieses Dokuments in eine andere Sprache, unter den obigen Bedingungen für veränderte Versionen zu kopieren und zu verteilen, mit der Ausnahme, dass diese Genehmigung übersetzt, statt im ursprünglichem Englisch, eingebunden werden kann, sofern diese Übersetzung (des Copyrights) von der Free Software Foundation genehmigt ist.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Autoren . . . 1

1.2 Wo Sie diese Anleitung bekommen (und verfügbare Formate) . . . 2

1.3 Organisatorisches / Feedback . . . 3

1.4 Vorwissen . . . 3

1.5 Dinge, die noch geschrieben werden müssen (FIXME/TODO) . . . 4

1.6 Änderungsübersicht/Changelog/Geschichte . . . 7

1.6.1 Version 3.10 (November 2006) . . . 7

1.6.2 Version 3.9 (Oktober 2006) . . . 7

1.6.3 Version 3.8 (Juli 2006) . . . 8

1.6.4 Version 3.7 (April 2006) . . . 8

1.6.5 Version 3.6 (März 2006) . . . 8

1.6.6 Version 3.5 (November 2005) . . . 9

1.6.7 Version 3.4 (August-September 2005) . . . 9

1.6.8 Version 3.3 (Juni 2005) . . . 10

1.6.9 Version 3.2 (März 2005) . . . 10

1.6.10 Version 3.1 (January 2005) . . . 11

1.6.11 Version 3.0 (December 2004) . . . 11

1.6.12 Version 2.99 (March 2004) . . . 11

1.6.13 Version 2.98 (December 2003) . . . 12

1.6.14 Version 2.97 (September 2003) . . . 12

1.6.15 Version 2.96 (august 2003) . . . 13

1.6.16 Version 2.95 (June 2003) . . . 13

1.6.17 Version 2.94 (April 2003) . . . 13
1.6.18 Version 2.93 (march 2003) . . . 14
1.6.19 Version 2.92 (February 2003) . . . 14
1.6.20 Version 2.91 (January/February 2003) . . . 14
1.6.21 Version 2.9 (December 2002) . . . 15
1.6.22 Version 2.8 (November 2002) . . . 15
1.6.23 Version 2.7 (October 2002) . . . 15
1.6.24 Version 2.6 (September 2002) . . . 16
1.6.25 Version 2.5 (September 2002) . . . 16
1.6.26 Version 2.5 (August 2002) . . . 16
1.6.27 Version 2.4 . . . 18
1.6.28 Version 2.3 . . . 18
1.6.29 Version 2.3 . . . 19
1.6.30 Version 2.2 . . . 19
1.6.31 Version 2.1 . . . 19
1.6.32 Version 2.0 . . . 20
1.6.33 Version 1.99 . . . 21
1.6.34 Version 1.98 . . . 21
1.6.35 Version 1.97 . . . 21
1.6.36 Version 1.96 . . . 22
1.6.37 Version 1.95 . . . 22
1.6.38 Version 1.94 . . . 22
1.6.39 Version 1.93 . . . 22
1.6.40 Version 1.92 . . . 22
1.6.41 Version 1.91 . . . 23
1.6.42 Version 1.9 . . . 23
1.6.43 Version 1.8 . . . 23
1.6.44 Version 1.7 . . . 24
1.6.45 Version 1.6 . . . 24
1.6.46 Version 1.5 . . . 24
1.6.47 Version 1.4 . . . 25

1.6.48 Version 1.3 . . . 25
1.6.49 Version 1.2 . . . 25
1.6.50 Version 1.1 . . . 25
1.6.51 Version 1.0 . . . 25
1.7 Danksagungen . . . 25
1.7.1 Danksagungen des Übersetzers . . . 26

2 Bevor Sie beginnen ... 29
2.1 Wofür möchten Sie dieses System benutzen? . . . 29
2.2 Seien Sie wachsam gegenüber generellen Sicherheitsproblemen! . . . 29
2.3 Wie geht Debian mit der Sicherheit um? . . . 32

3 Vor und während der Installation 35
3.1 Setzen Sie ein Passwort im BIOS . . . 35
3.2 Partitionieren des Systems . . . 35
3.2.1 Wählen Sie eine sinnvolle Partitionierung . . . 35
3.3 Gehen Sie nicht ins Internet, bevor Sie nicht bereit sind . . . 37
3.4 Setzen Sie ein Passwort für root . . . 38
3.5 Aktivieren Sie Shadow-Passwörter und MD5-Passwörter . . . 38
3.6 Lassen Sie so wenige Dienste wie möglich laufen . . . 39
3.6.1 Daemons abschalten . . . 40
3.6.2 Abschalten von `inetd` oder seinen Diensten . . . 41
3.7 Installieren Sie möglichst wenig Software . . . 42
3.7.1 Entfernen von Perl . . . 44
3.8 Lesen Sie Debians Sicherheits-Mailinglisten . . . 46

4 Nach der Installation 47
4.1 Abonnement der Security-Announce-Mailingliste von Debian . . . 47
4.2 Ausführen von Sicherheitsupdates . . . 48
4.2.1 Sicherheitsaktualisierungen von Bibliotheken . . . 49
4.2.2 Sicherheitsaktualisierung des Kernels . . . 50
4.3 Änderungen im BIOS (noch einmal) . . . 51

4.4 Ein Passwort für LILO oder GRUB einstellen . . . 51
4.5 Entfernen des Root-Prompts von Initramfs . . . 52
4.6 Entfernen des Root-Promptes aus dem Kernel . . . 53
4.7 Booten von Diskette verbieten . . . 54
4.8 Einschränkender Konsolen-Zugang . . . 54
4.9 System-Neustart von der Konsole aus einschränken . . . 55
4.10 Partitionen auf die richtige Art einbinden . . . 56
4.10.1 `/tmp` noexec setzen . . . 57
4.10.2 Setzen von /usr auf nur-lesen . . . 57
4.11 Den Benutzern einen sicheren Zugang bereitstellen . . . 58
4.11.1 Nutzerauthentifizierung: PAM . . . 58
4.11.2 Ressourcen-Nutzung limitieren: Die Datei `limits.conf` . . . 61
4.11.3 Aktionen bei der Benutzeranmeldung: Editieren von `/etc/login.defs` 63
4.11.4 ftp Einschränken: Editieren von `/etc/ftpusers` . . . 65
4.11.5 Verwendung von su . . . 65
4.11.6 Verwendung von sudo . . . 65
4.11.7 Administrativen Fernzugriff verweigern . . . 65
4.11.8 Den Nutzerzugang einschränken . . . 66
4.11.9 Überprüfen der Nutzer . . . 66
4.11.10 Nachprüfung der Nutzerprofile . . . 68
4.11.11 umasks der Nutzer einstellen . . . 69
4.11.12 Nutzer Sicht/Zugriff limitieren . . . 70
4.11.13 Erstellen von Benutzerpasswörtern . . . 71
4.11.14 Kontrolle der Benutzerpasswörter . . . 72
4.11.15 Ausloggen von untätigen Nutzern . . . 72
4.12 Die Nutzung von tcpwrappers . . . 73
4.13 Die Wichtigkeit von Logs und Alarmen . . . 74
4.13.1 Nutzung und Anpassung von `logcheck` . . . 75
4.13.2 Konfiguration, wohin Alarmmeldungen geschickt werden . . . 76
4.13.3 Nutzen eines loghosts . . . 77
4.13.4 Zugriffsrechte auf Log-Dateien . . . 78

4.14 Den Kernel patchen . . . 78
4.15 Schutz vor Pufferüberläufen . . . 80
4.15.1 Kernelpatch zum Schutz vor Pufferüberläufen . . . 81
4.15.2 Schutz durch `libsafe` . . . 82
4.15.3 Prüfprogramme für Pufferüberläufe . . . 82
4.16 Sichere Übertragung von Dateien . . . 82
4.17 Einschränkung und Kontrolle des Dateisystems . . . 83
4.17.1 Benutzung von Quotas . . . 83
4.17.2 Die für das ext2-Dateisystem spezifischen Attribute (chattr/lsattr) . . . 84
4.17.3 Prüfung der Integrität des Dateisystems . . . 85
4.17.4 Aufsetzen einer Überprüfung von setuid . . . 86
4.18 Absicherung des Netzwerkzugangs . . . 86
4.18.1 Konfiguration der Netzwerkfähigkeiten des Kernels . . . 87
4.18.2 Konfiguration von Syncookies . . . 87
4.18.3 Absicherung des Netzwerks beim Hochfahren . . . 88
4.18.4 Konfiguration der Firewall . . . 92
4.18.5 Lösung des Problems der Weak-End-Hosts . . . 92
4.18.6 Schutz vor ARP-Angriffen . . . 93
4.19 Einen Schnappschuss des Systems erstellen . . . 94
4.20 Andere Empfehlungen . . . 96
4.20.1 Benutzen Sie keine Software, die von svgalib abhängt . . . 96

5 Absichern von Diensten, die auf Ihrem System laufen 97
5.1 Absichern von ssh . . . 98
5.1.1 SSH in ein Chroot-Gefängnis einsperren . . . 100
5.1.2 Ssh-Clients . . . 100
5.1.3 Verbieten von Dateitransfers . . . 100
5.2 Absichern von Squid . . . 100
5.3 Absichern von FTP . . . 102
5.4 Zugriff auf das X-Window-System absichern . . . 103
5.4.1 Überprüfen Ihres Display-Managers . . . 104

5.5 Absichern des Druckerzugriffs (die lpd- und lprng-Problematik) . . . 105
5.6 Absichern des Mail-Dienstes . . . 106
5.6.1 Konfiguration eines Nullmailers . . . 106
5.6.2 Anbieten von sicherem Zugang zu Mailboxen . . . 108
5.6.3 Sicherer Empfang von Mails . . . 109
5.7 Sichern von BIND . . . 109
5.7.1 Bind-Konfiguration um Missbrauch zu verhindern . . . 110
5.7.2 Ändern des BIND-Benutzers . . . 112
5.7.3 Chroot-Gefängnis für Name-Server . . . 114
5.8 Absichern von Apache . . . 116
5.8.1 Verhindern, dass Benutzer Web-Inhalte veröffentlichen . . . 117
5.8.2 Rechte von Log-Datei . . . 117
5.8.3 Veröffentlichte Web-Dateien . . . 118
5.9 Absichern von Finger . . . 118
5.10 Allgemeine chroot- und suid-Paranoia . . . 118
5.10.1 Automatisches Erstellen von Chroot-Umgebungen . . . 119
5.11 Allgemeine Klartextpasswort-Paranoia . . . 120
5.12 NIS deaktivieren . . . 120
5.13 Sichern von RPC-Diensten . . . 120
5.13.1 Vollständiges Deaktivieren von RPC-Diensten . . . 121
5.13.2 Einschränken des Zugriffs auf RPC-Dienste . . . 121
5.14 Hinzufügen von Firewall-Fähigkeiten . . . 122
5.14.1 Firewallen des lokalen Systems . . . 122
5.14.2 Schützen anderer Systeme durch eine Firewall . . . 123
5.14.3 Aufsetzen einer Firewall . . . 123

6 Automatisches Abhärten von Debian-Systemen 133

6.1 Harden . . . 133
6.2 Bastille Linux . . . 135

7 Die Infrastruktur für Sicherheit in Debian **137**

7.1 Das Sicherheitsteam von Debian . . . 137

7.2 Debian-Sicherheits-Ankündigungen . . . 138

7.2.1 Querverweise der Verwundbarkeiten . . . 139

7.2.2 CVE-Kompatibilität . . . 139

7.3 Die Infrastruktur der Sicherheit bei der Paketerstellung in Debian . . . 140

7.3.1 Leitfaden über Sicherheitsaktualisierungen für Entwickler . . . 142

7.4 Paketsignierung in Debian . . . 145

7.4.1 Der vorgeschlagene Plan zur Prüfung von Paketsignaturen . . . 145

7.4.2 Secure Apt . . . 146

7.4.3 Überprüfung der Distribution mit der `Release`-Datei . . . 147

7.4.4 Prüfung von Debian-fremden Quellen . . . 161

7.4.5 Alternativer Entwurf zur Einzelsignierung von Paketen . . . 161

8 Sicherheitswerkzeuge in Debian **163**

8.1 Programme zur Fernprüfung der Verwundbarkeit . . . 163

8.2 Werkzeuge zum Scannen von Netzwerken . . . 164

8.3 Interne Prüfungen . . . 165

8.4 Testen des Quellcodes . . . 165

8.5 Virtual Private Networks (virtuelle private Netzwerke) . . . 165

8.5.1 Point-to-Point-Tunneling . . . 166

8.6 Infrastruktur für öffentliche Schlüssel (Public Key Infrastructure, PKI) . . . 167

8.7 SSL Infrastruktur . . . 168

8.8 Antiviren-Programme . . . 168

8.9 GPG-Agent . . . 170

9 Der gute Umgang von Entwicklern mit der Sicherheit des OS **173**

9.1 Das richtige Vorgehen für die Nachprüfung der Sicherheit und Gestaltung . . . 173

9.2 Benutzer und Gruppen für Daemons erstellen . . . 175

10 Vor der Kompromittierung **179**

10.1 Halten Sie Ihr System sicher . . . 179

10.1.1 Beobachtung von Sicherheitslücken . . . 179

10.1.2 Fortlaufende Aktualisierung des Systems . . . 180

10.1.3 Vermeiden Sie den Unstable-Zweig . . . 184

10.1.4 Sicherheitsunterstützung für den Testing-Zweig . . . 184

10.1.5 Automatische Aktualisierungen in einem Debian GNU/Linux System . . . 185

10.2 Periodische Überprüfung der Integrität . . . 187

10.3 Aufsetzen einer Eindringlingserkennung . . . 187

10.3.1 Netzwerk basierende Eindringlingserkennung . . . 188

10.3.2 Host basierende Eindringlingserkennung . . . 189

10.4 Vermeiden von Root-Kits . . . 189

10.4.1 Ladbare Kernel-Module (LKM) . . . 189

10.4.2 Erkennen von Root-Kits . . . 190

10.5 Geniale/paranoide Ideen — was Sie tun können . . . 191

10.5.1 Aufstellen eines Honigtopfes (honeypot) . . . 192

11 Nach einer Kompromittierung (Reaktion auf einem Vorfall) **195**

11.1 Allgemeines Verhalten . . . 195

11.2 Anlegen von Sicherheitskopien Ihres Systems . . . 196

11.3 Setzen Sie sich mit dem lokal CERT in Verbindung . . . 197

11.4 Forensische Analyse . . . 197

12 Häufig gestellte Fragen / Frequently asked Questions (FAQ) **199**

12.1 Sicherheit im Debian Betriebssystem . . . 199

12.1.1 Ist Debian sicherer als X? . . . 199

12.1.2 In Bugtraq gibt es viele Debian-Fehler. Heißt das, dass es sehr gefährdet ist? . . . 200

12.1.3 Hat Debian irgendein Zertifikat für Sicherheit? . . . 201

12.1.4 Gibt es irgendein Abhärtungsprogramm für Debian? . . . 201

12.1.5 Ich möchte einen XYZ-Dienst laufen lassen. Welchen sollte ich benutzen? 201

12.1.6 Wie mache ich den Dienst XYZ unter Debian sicherer? . . . 202

12.1.7 Wie kann ich die Banner der Dienste entfernen? . . . 202
12.1.8 Sind alle Debian Pakete sicher? . . . 202
12.1.9 Warum sind einige Log- und Konfigurationsdateien für die Welt lesbar? Ist das nicht unsicher? . . . 203
12.1.10 Warum hat /root/ (oder NutzerX) die Rechte 755? . . . 203
12.1.11 Nach der Installation von grsec oder einer Firewall bekomme ich viele Nachrichten auf der Konsole. Wie entferne ich sie? . . . 204
12.1.12 Benutzer und Gruppen des Betriebssystems . . . 204
12.1.13 Warum gibt es eine neue Gruppe, wenn ich einen neuen Nutzer anlege? (Oder warum gibt Debian jedem Nutzer eine eigene Gruppe?) . . . 208
12.1.14 Fragen über Dienste und offene Ports . . . 209
12.1.15 Allgemeine Sicherheitsfragen . . . 211
12.1.16 Wie muss ich vorgehen, wenn ich meinen Nutzern einen Dienst anbieten möchte, ihnen aber kein Shell-Konto geben will? . . . 212
12.2 Mein System ist angreifbar! (Sind Sie sich sicher?) . . . 213
12.2.1 Der Scanner X zur Einschätzung der Verwundbarkeit sagt, dass mein Debian-System verwundbar wäre? . . . 213
12.2.2 Ich habe in meinen Logfiles einen Angriff gesehen: Ist mein System kompromittiert? . . . 213
12.2.3 Ich habe in meinen Logs merkwürdige “MARK”-Einträge gefunden. Wurde ich gehackt? . . . 213
12.2.4 Ich habe Nutzer gefunden, die laut meinen Logfiles ‘su’ benutzen: Bin ich kompromittiert? . . . 214
12.2.5 Ich habe ‘possible SYN flooding’ in meinen Logs entdeckt: Werde ich angegriffen? . . . 214
12.2.6 Ich habe seltsame Root-Sessions in meinen Logs entdeckt: Wurde ich gehackt? . . . 215
12.2.7 Ich bin Opfer eines Einbruchs, was soll ich jetzt tun? . . . 215
12.2.8 Wie verfolge ich einen Angriff zurück? . . . 216
12.2.9 Das Programm X in Debian ist angreifbar – was soll ich tun? . . . 216
12.2.10 Laut der Versionsnummer eines Paketes läuft bei mir immer noch eine angreifbare Version! . . . 216
12.2.11 Spezielle Software . . . 216
12.3 Fragen zum Sicherheitsteam von Debian . . . 217

12.3.1 Was ist eine Debian-Sicherheits-Ankündigung (Debian Security Advisory, DSA)? . . . 217
12.3.2 Die digitale Signatur eines Debian-Anweisung wird nicht korrekt verifiziert! . . . 217
12.3.3 Wie wird die Sicherheit in Debian gehandhabt? . . . 217
12.3.4 Wieso spielen Sie mit einer alten Version des Pakets herum? . . . 218
12.3.5 Was sind die Richtlinien für ein repariertes Paket, um auf security.debian.org zu erscheinen? . . . 218
12.3.6 Die Versionsnummer für ein Paket zeigt an, dass ich immer noch eine verwundbare Version verwende! . . . 218
12.3.7 Wie wird die Sicherheit für Testing und Unstable gehandhabt? . . . 219
12.3.8 Ich verwende eine ältere Version von Debian. Wird sie vom Debian-Sicherheitsteam unterstützt? . . . 219
12.3.9 Warum gibt es keinen offiziellen Mirror von security.debian.org? . . . 219
12.3.10 Ich habe DSA 100 und DSA 102 gesehen, doch wo ist DSA 101? . . . 219
12.3.11 Wie kann ich das Sicherheitsteam erreichen? . . . 220
12.3.12 Was ist der Unterschied zwischen security@debian.org und debian-security@lists.debian.org? . . . 220
12.3.13 Wie kann ich das Debian-Sicherheitsteam unterstützen? . . . 220
12.3.14 Aus wem setzt sich das Sicherheitsteam zusammen? . . . 221
12.3.15 Prüft das Debian-Sicherheitsteam jedes Paket in Debian? . . . 221
12.3.16 Wie lange braucht Debian, um die Angriffs-Möglichkeit XXXX zu reparieren? . . . 221

A Der Abhärtungsprozess Schritt für Schritt **223**

B Checkliste der Konfiguration **227**

C Aufsetzen eines autonomen IDS **231**

D Aufsetzen einer überbrückenden Firewall (bridge Firewall) **237**
D.1 Eine Bridge mit NAT- und Firewall-Fähigkeiten . . . 238
D.2 Eine Bridge mit Firewall-Fähigkeiten . . . 238
D.3 Grundlegende Iptables-Regeln . . . 239

E Beispielskript, um die Standard-Installation von Bind zu ändern **241**

F Schutz der Sicherheitsaktualisierung durch eine Firewall **247**

G `Chroot`-Umgebung für `SSH` **249**

G.1 SSH-Benutzer in ein Chroot-Gefängnis einsperren 249

G.1.1 Einsatz von `libpam-chroot` 250

G.1.2 Patchen des `ssh`-Servers 252

G.2 Einsperren des SSH-Servers in einem Chroot-Gefängnis 254

G.2.1 Einrichten eines minimalen Systems (der wirklich leichte Weg) 254

G.2.2 Automatisches Erstellen der Umgebung (der leichte Weg) 255

G.2.3 Die Chroot-Umgebung von Hand erstellen (der schwierige Weg) 260

H `Chroot`-Umgebung für `Apache` **267**

H.1 Einleitung 267

H.1.1 Lizenz 267

H.2 Installation des Servers 267

H.3 Weiterführende Informationen 272

Kapitel 1

Einleitung

Eines der schwierigsten Dinge beim Schreiben über Sicherheit ist, dass jeder Fall einzigartig ist. Sie müssen zwei Dinge beachten: Die Gefahr aus der Umgebung und das Bedürfnis an Sicherheit Ihrer Seite, Ihres Hosts oder Ihres Netzwerkes. So unterscheiden sich zum Beispiel die Sicherheitsbedürfnisse eines Heimanwenders komplett von den Sicherheitsbedürfnissen des Netzwerkes einer Bank. Während die Hauptgefahr eines Heimanwenders von "Script-Kiddies" ausgeht, muss sich das Netzwerk einer Bank um direkte Angriffe sorgen. Zusätzlich muss eine Bank die Daten ihrer Kunden mit mathematischer Präzision beschützen. Um es kurz zu machen: Jeder Nutzer muss selbst zwischen Benutzerfreundlichkeit und Sicherheit/Paranoia abwägen.

Beachten Sie bitte, dass diese Anleitung nur Software-Themen behandelt. Die beste Software der Welt kann Sie nicht schützen, wenn jemand direkten Zugang zu Ihrem Rechner hat. Sie können ihn unter Ihren Schreibtisch stellen, oder Sie können ihn in einen starken Bunker mit einer ganzen Armee davor stellen. Trotzdem kann der Rechner unter Ihrem Schreibtisch weitaus sicherer sein – von der Software-Seite aus gesehen – als der eingebunkerte, wenn Ihr Schreibtisch-Rechner richtig konfiguriert und die Software des eingebunkerten Rechners voller Sicherheitslöcher ist. Sie müssen beide Möglichkeiten betrachten.

Dieses Dokument gibt Ihnen lediglich einen kleinen Überblick, was Sie tun können, um die Sicherheit Ihres Debian GNU/Linux Systems zu erhöhen. Wenn Sie bereits andere Dokumente über Sicherheit unter Linux gelesen haben, werden Sie feststellen, dass es einige Überschneidungen geben wird. Wie auch immer: Dieses Dokument versucht nicht, die ultimative Informationsquelle zu sein, es versucht nur, die gleichen Informationen so zu adaptieren, dass sie gut auf ein Debian GNU/Linux System passen. Unterschiedliche Distributionen erledigen manche Dinge auf unterschiedliche Art (zum Beispiel den Aufruf von Daemons); hier finden Sie Material, das zu Debians Prozeduren und Werkzeugen passt.

1.1 Autoren

Der aktuelle Betreuer dieses Dokuments ist Javier Fernández-Sanguino Peña (mailto:jfs@debian.org). Falls Sie Kommentare, Ergänzungen oder Vorschläge haben, schicken Sie ihm

diese bitte. Sie werden dann in künftigen Ausgaben dieses Handbuchs berücksichtigt werden.

Dieses Handbuch wurde als *HOWTO* von Alexander Reelsen (mailto:ar@rhwd.de) ins Leben gerufen. Nachdem es im Internet veröffentlicht wurde, gliederte es Javier Fernández-Sanguino Peña (mailto:jfs@debian.org) in das Debian-Dokumentations-Projekt (http://www.debian.org/doc) ein. Zahlreiche Menschen haben etwas zu diesem Handbuch beigesteuert (alle Beiträge sind im Changelog aufgeführt), aber die folgenden haben gesonderte Erwähnung verdient, da sie bedeutende Beiträge geleistet haben (ganze Abschnitte, Kapitel oder Anhänge):

- Stefano Canepa
- Era Eriksson
- Carlo Perassi
- Alexandre Ratti
- Jaime Robles
- Yotam Rubin
- Frederic Schutz
- Pedro Zorzenon Neto
- Oohara Yuuma
- Davor Ocelic

Bei Fehlern in dieser Übersetzung wenden Sie sich bitte an den aktuellen deutschen Übersetzer Simon Brandmair (mailto:sbrandmair@gmx.net) oder (wenn dieser nicht erreichbar ist) an die Mailingliste (mailto:debian-l10n-german@lists.debian.org).[1]

1.2 Wo Sie diese Anleitung bekommen (und verfügbare Formate)

Sie können sich die neueste Version der "Anleitung zum Absichern von Debian" beim Debian-Dokumentationsprojekt (http://www.de.debian.org/doc/manuals/securing-debian-howto/) herunterladen oder anschauen. Wenn Sie eine Kopie von einer anderen Seite lesen, überprüfen Sie bitte die Hauptversion, ob sie neue Informationen enthält. Wenn Sie eine Übersetzung lesen (was Sie im Moment tun, d.Ü.), vergleichen Sie bitte die Version der Übersetzung mit der neuesten Version. Falls Sie feststellen, dass die Übersetzung veraltet ist, sollten Sie in Betracht ziehen, die Originalversion zu verwenden oder zumindest

[1] Es würde eine Menge Arbeit ersparen, wenn die Verbesserungen in die SGML-Dateien eingearbeitet werden. Diese sind mittels CVS abrufbar und können auch über die Webschnittstelle (http://cvs.debian.org/ddp/manuals.sgml/securing-howto/?cvsroot=debian-doc) abgerufen werden. d.Ü.

die ‘Änderungsübersicht/Changelog/Geschichte’ auf Seite 7 durchsehen, um zu wissen, was geändert wurde.

Wenn Sie eine vollständige Kopie des Handbuchs wollen, können Sie die Text-Version (http://www.de.debian.org/doc/manuals/securing-debian-howto/securing-debian-howto.de.txt) oder die PDF-Version (http://www.de.debian.org/doc/manuals/securing-debian-howto/securing-debian-howto.de.pdf) von der Seite des Debian-Dokumentationsprojektes herunterladen. Diese Versionen können sinnvoller sein, wenn Sie das Dokument auf ein tragbares Medium kopieren oder ausdrucken wollen. Seien Sie aber gewarnt, das Dokument ist über 200 Seiten lang und einige Abschnitte von Code ist in der PDF-Version wegen den eingesetzten Formatierungswerkzeugen nicht richtig umgebrochen und könnte daher nur unvollständig ausgedruckt werden.

Das Dokument ist auch in den Formaten HTML, txt und PDF im Paket harden-doc (http://packages.debian.org/harden-doc) enthalten. Beachten Sie allerdings, dass das Paket nicht genauso aktuell sein muss wie das Dokument, das Sie auf der Debian-Seite finden (Sie können sich aber immer eine aktuelle Version aus dem Quellpaket bauen).

Sie können auch die Veränderungen durchforsten, die am Dokument vorgenommen wurden, indem Sie die Protokolle der Versionskontrolle mit dem CVS-Server (http://cvs.debian.org/ddp/manuals.sgml/securing-howto/?cvsroot=debian-doc) durchsehen.

1.3 Organisatorisches / Feedback

Nun kommt der offizielle Teil. Derzeit sind die meisten Teile dieser Anleitung noch von mir (Alexander Reelsen) geschrieben, aber meiner Meinung nach sollte dies nicht so bleiben. Ich wuchs mit freier Software auf und lebe mit ihr, sie ist ein Teil meiner alltäglichen Arbeit und ich denke, auch von Ihrer. Ich ermutige jedermann, mir Feedback, Tipps für Ergänzungen oder andere Vorschläge, die Sie haben könnten, zuzuschicken.

Wenn Sie denken, dass Sie einen bestimmten Abschnitt oder Paragraphen besser pflegen können, dann schreiben Sie dem Dokumenten-Betreuer und Sie dürfen es gerne erledigen. Insbesondere, wenn Sie eine Stelle finden, die mit "FIXME" markiert wurde – was bedeutet, dass die Autoren noch nicht die Zeit hatten oder sich noch Wissen über das Thema aneignen müssen – schicken Sie ihnen sofort eine E-Mail.

Für diese Anleitung ist es natürlich äußerst wichtig, dass sie weiter gepflegt und auf dem neusten Stand gehalten wird. Auch Sie können Ihren Teil dazu beitragen. Bitte unterstützen Sie uns.

1.4 Vorwissen

Die Installation von Debian GNU/Linux ist nicht wirklich schwer, und Sie sollten in der Lage gewesen sein, es zu installieren. Wenn Ihnen andere Linux-Distributionen, Unixe oder die grundsätzliche Sicherheitskonzepte ein wenig vertraut sind, wird es Ihnen leichter fallen, diese Anleitung zu verstehen, da nicht auf jedes winzige Detail eingegangen werden kann (oder

dies wäre ein Buch geworden und keine Anleitung). Wenn Sie jedoch mit diesen Dingen noch nicht so vertraut sind, sollten Sie vielleicht einen Blick in die 'Seien Sie wachsam gegenüber generellen Sicherheitsproblemen!' auf Seite 29 für tiefer gehende Informationen werfen.

1.5 Dinge, die noch geschrieben werden müssen (FIXME/TODO)

Dieses Kapitel beschreibt die Dinge, welche in diesem Handbuch noch verbessert werden müssen. Einige Abschnitte beinhalten *FIXME*- oder *TODO*-Markierungen, in denen beschrieben wird, welche Dinge fehlen (oder welche Aufgaben erledigt werden müssen). Der Zweck dieses Kapitels ist es, die Dinge, die zukünftig in dieses Handbuch aufgenommen werden könnten, und die Verbesserungen, die durchgeführt werden müssen (oder bei denen es interessant wäre, sie einzufügen) zu beschreiben.

Wenn Sie glauben, dass Sie Hilfe leisten könnten, den auf dieser Liste aufgeführten Punkten (oder solchem im Text) abzuhelfen, setzen Sie sich mit dem Hauptautor ('Autoren' auf Seite 1) in Verbindung.

- Erweiterung der Informationen zur Reaktion auf einen Zwischenfall, unter Umständen auch mit einigen Vorschlägen von Red Hats Sicherheitsanleitung chapter on incident response (`http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/security-guide/ch-response.html`).
- Vorstellen von entfernten Überwachungswerkzeugen (um die Erreichbarkeit des Systems zu überprüfen) wie `monit`, `daemontools` und `mon`. Vergleiche `http://linux.oreillynet.com/pub/a/linux/2002/05/09/sysadminguide.html`.
- Überprüfung, ob `http://www.giac.org/practical/gsec/Chris_Koutras_GSEC.pdf` wichtige Informationen hat, die hier noch nicht behandelt werden.
- Informationen, wie man einen Laptop mit Debian einrichtet `http://www.giac.org/practical/gcux/Stephanie_Thomas_GCUX.pdf`.
- Informationen, wie man unter Debian GNU/Linux eine Firewall aufsetzt. Der Firewalls betreffende Abschnitt orientiert sich derzeit an Einzelplatz-Systemen (die keine anderen System schützen müssen); auch auf das Testen des Setups eingehen.
- Wie man eine Proxy-Firewall unter Debian GNU/Linux aufsetzt, unter der Angabe, welche Pakete Proxy-Dienste anbieten (zum Beispiel `xfwp`, `ftp-proxy`, `redir`, `smtpd`, `dnrd`, `jftpgw`, `oops`, `pdnsd`, `perdition`, `transproxy`, `tsocks`). Sollte zu einer Anleitung mit weiteren Informationen verweisen. Erwähnenswert ist, dass `zorp` jetzt Teil von Debian ist und eine Proxy-Firewall *ist* (und auch der Programmautor Debian-Pakete zur Verfügung stellt.)
- Informationen über die Service-Konfiguration mit file-rc
- Alle Referenzen und URLs prüfen und die nicht mehr verfügbaren aktualisieren oder entfernen

- Informationen über möglichen Ersatz (unter Debian) für häufig eingesetzte Server, die bei eingeschränktem Funktionsumfang nützlich sind. Beispiele:
  - lokaler lpr mit cups (Paket)?
  - remote lrp mit lpr
  - bind mit dnrd/maradns
  - apache mit dhttpd/thttpd/wn (tux?)
  - exim/sendmail mit ssmtpd/smtpd/postfix
  - squid mit tinyproxy
  - ftpd mit oftpd/vsftp
  - ...
- Mehr Informationen über die sicherheitsrelevanten Patches des Kernels unter Debian einschließlich der oben aufgeführten, und insbesondere wie man diese Patches unter einem Debian-System benutzt.
  - Erkennung von Eindringlingen (Linux Intrusion Detection `kernel-patch-2.4-lids`)
  - Linux Trustees (im Paket `trustees`)
  - NSA Enhanced Linux (`http://wiki.debian.org/SELinux`)
  - kernel-patch-2.2.18-openwall (`http://packages.debian.org/kernel-patch-2.2.18-openwall`)
  - `kernel-patch-freeswan`, `kernel-patch-int`
- Details, wie man unnötige Netzwerkdienste deaktiviert (abgesehen von `inetd`), dies ist teilweise Teil des Abhärtungsprozesses, könnte aber etwas ausgeweitet werden.
- Informationen über Passwort-Rotation, was sehr nah an grundsätzliche Regeln (Policies) herankommt
- Policies und die Aufklärung der Nutzer über die Policy
- Mehr über tcpwrapper und wrapper im Allgemeinen?
- `hosts.equiv` und andere wichtige Sicherheitslöcher
- mögliche Probleme bei der Dateifreigabe, wie Samba und NFS?
- suidmanager/dpkg-statoverrides.
- lpr und lprng.
- Abschalten der GNOME-IP-Dinge.

- Erwähnen von pam_chroot (siehe http://lists.debian.org/debian-security/2002/debian-security-200205/msg00011.html) und seine Nützlichkeit, um Nutzer einzuschränken. Einführende Informationen in Verbindung mit http://online.securityfocus.com/infocus/1575. pdmenu sind zum Beispiel bereits unter Debian verfügbar (während flash das nicht ist).
- Darüber reden, Dienste mit einer chroot-Umgebung zu versehen, mehr Informationen dazu unter http://www.linuxfocus.org/English/January2002/article225.shtml, http://www.nuclearelephant.com/papers/chroot.html und http://www.linuxsecurity.com/feature_stories/feature_story-99.html.
- Programme erwähnen, die Chroot-Gefängnisse (chroot jails) herstellen. compartment und chrootuid warten noch in incoming. Einige andere (makejail, jailer) könnten ebenfalls eingeführt werden.
- Die Informationen von Pedro Zornenon über das Einsperren per chroot von Bind 8, aber leider nur für Potato :(, siehe http://people.debian.org/~pzn/howto/chroot-bind.sh.txt (einfügen des ganzen Skriptes?).
- Mehr Informationen über Software zur Analyse von Protokoll-Dateien (log-Dateien, logs; zum Beispiel logcheck und logcolorise).
- "Fortgeschrittenes" Routing (Traffic-Regelungen sind sicherheitsrelevant)
- Zugang über SSH so einschränken, dass man nur bestimmte Kommandos ausführen kann
- Die Benutzung von dpkg-statoverride.
- Sichere Wege, mehreren Nutzern den Zugriff auf CD-Brenner zu erlauben
- Sichere Wege, um Sound zusammen mit einem Display über ein Netzwerk zu leiten (so dass die Sounds eines X-Clients über die Hardware eines X-Servers abgespielt werden)
- Absichern von Web-Browsern
- Aufsetzen von ftp über ssh
- Die Benutzung von verschlüsselten Loopback-Dateisystemen
- Verschlüsselung eines ganzen Dateisystems
- Steganographie-Tools
- Aufsetzen eines PKA für eine Organisation
- LDAP benutzen zur Verwaltung der User. Es gibt ein HOWTO zu ldap+kerberos für Debian auf http://www.bayour.com von Turbo Fredrikson.
- Wie man Informationen mit begrenztem Nutzen wie z.B. /usr/share/doc oder /usr/share/man auf Produktivsystemen entfernt (jawohl, security by obscurity).

- Mehr Informationen über lcap, die sich auf die README-Datei des Pakets stützen (gut, die Datei ist noch nicht vorhaben, vergleiche Bug #169465 (http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=169465)) und diesen Artikel von LWN: Kernel development (http://lwn.net/1999/1202/kernel.php3).
- Füge Colins Artikel hinzu, wie man eine Chroot-Umgebung für ein komplettes Sid-System aufsetzt (http://people.debian.org/~walters/chroot.html).
- Informationen darüber, wie man mehrere `snort`-Sensoren in einem System betreibt (beachte die Fehlerberichte zu `snort`).
- Informationen, wie man einen Honigtopf (honeypot) einrichtet (`honeyd`)
- Darstellung der Situation von FreeSwan (verwaist) und OpenSwan. Der Abschnitt über VPN muss überarbeitet werden.
- Füge einen gesonderten Abschnitt über Datenbanken hinzu, ihre Standardwerte und, wie man den Zugriff absichert.
- Füge einen Abschnitt über den Nutzen von virtuellen Servern (wie Xen u.a.) hinzu.
- Erkläre, wie Programme zur Überprüfung der Integrität verwendet werden (AIDE, integrit oder samhain). Die Grundlagen sind einfach und könnten auch einige Verbesserungen der Konfiguration enthalten.

1.6 Änderungsübersicht/Changelog/Geschichte

1.6.1 Version 3.10 (November 2006)

Änderung von Javier Fernández-Sanguino Peña

- Gab Beispiele, wie rdepends von apt-cache verwendet wird. Wurde von Ozer Sarilar vorgeschlagen.
- Korrigierte den Verweis auf das Benutzerhandbuch von Squid auf Grund seines Umzugs. Wurde von Oskar Pearson (dem Herausgeber) mitgeteilt.
- Korrigierte Informationen über umask, seine logins.defs (nicht limits.conf), wo dies für einen Anmeldungen konfiguriert werden kann. Stellte den Standard von Debian und strengere Werte für Benutzer und Root dar. Vielen Dank an Reinhard Tartler für das Auffinden des Fehlers.

1.6.2 Version 3.9 (Oktober 2006)

Änderung von Javier Fernández-Sanguino Peña

- Fügte Informationen hinzu, wie man Sicherheitslücken verfolgt. Hinweis auf den Debian-Testing-Sicherheits-Tracker.
- Fügte weitere Informationen über die Sicherheitsunterstützung für Testing hinzu.
- Korrigierte eine große Anzahl von Tippfehlern mit einem Patch von Simon Brandmair.
- Fügte einen Abschnitt hinzu, wie das Root-Prompt bei initramfs abgestellt wird. Wurde von Max Attems beigesteuert.
- Entfernte Verweise auf queso.
- Hinweis in der Einleitung, dass nun auch Testing vom Sicherheitsteam unterstützt wird.

1.6.3 Version 3.8 (Juli 2006)

Änderung von Javier Fernández-Sanguino Peña

- Schrieb die Hinweise neu, wie man SSH in einer Chroot-Umgebung einsperrt, um die verschiedenen Optionen deutlicher herauszustellen. Vielen Dank an Bruce Park, der auf verschiedene Fehler in diesem Anhang hinwies.
- Verbesserte den Aufruf von lsof, wie es von Christophe Sahut vorgeschlagen wurde.
- Fügte einen Patch von Uwe Hermann zur Verbesserung von Tippfehlern ein.
- Verbesserte einen Tippfehler, der von Moritz Naumann entdeckt wurde.

1.6.4 Version 3.7 (April 2006)

Änderung von Javier Fernández-Sanguino Peña

- Fügte einen Abschnitt über den besten Umgang der Entwickler von Debian mit Sicherheitsfragen hinzu.
- Fügte eine Firewall-Skript mit Kommentaren von WhiteGhost an.

1.6.5 Version 3.6 (März 2006)

Änderung von Javier Fernández-Sanguino Peña

- Fügte einen Patch von Thomas Sjögren ein, der beschreibt, dass `noexec` wie erwartet mit »neuen« Kernel arbeitet, der Informationen über den Umgang mit temporären Dateien hinzufügt und einige Verweise auf externe Dokumentationen.

- Fügte nach einem Vorschlag von Freek Dijkstra einen Verweis auf Dan Farmers und Wietse Venemas Webseite über forensische Entdeckungen ein und erweiterte den Abschnitt über forensische Analyse etwas mit weiteren Verweisen.
- Verbesserte dank Christoph Auer die URL des italienischen CERT.
- Verwendete wieder Joey Hess' Informationen aus dem Wiki über Secure Apt und fügte sie in den Infrastrukturabschnitt ein.

1.6.6 Version 3.5 (November 2005)

Änderung von Javier Fernández-Sanguino Peña

- Hinweis im SSH-Abschnitt, dass chroot nicht funktioniert, wenn die Option nodev mit der Partition verwendet wird, und auf das aktuelle ssh-Paket mit dem chroot-Patch. Vielen Dank an Lutz Broedel für diese Hinweise.
- Ausbesserung eines Tippfehlers, der von Marcos Roberto Greiner entdeckt wurde (md5sum sollte sha1sum im Code-Schnipsel sein)
- Fügte Jens Seidels Patch ein, der eine Anzahl von Paketnamen und Tippfehlern verbesserte.
- Kleine Aktualisierung des Werkzeugabschnitts, entfernte Werkzeuge, die nicht länger verfügbar sind, und fügte einige neue hinzu.
- Schrieb Teile des Abschnitts neu, in dem es darum geht, wo und in welchen Formaten dieses Dokument erhältlich ist (die Webseite stellt eine PDF-Version zur Verfügung). Merkte auch an, dass Kopien auf anderen Seiten und Übersetzungen veraltet sein könnten (viele der Treffer auf Google für dieses Handbuch auf anderen Seiten sind veraltet).

1.6.7 Version 3.4 (August-September 2005)

Änderung von Javier Fernández-Sanguino Peña

- Verbesserte die Erhöhung der Sicherheit nach der Installation im Zusammenhang mit der Kernelkonfiguration für den Schutz der Netzwerkebene mit der Datei sysctl.conf. Wurde von Will Moy zur Verfügung gestellt.
- Verbesserte den Abschnitt über gdm mit Hilfe von Simon Brandmair.
- Ausbesserungen von Tippfehlern, die von Frédéric Bothamy und Simon Brandmair entdeckt wurden.
- Verbesserungen im Abschnitt "Nach der Installation" im Zusammenhang, wie MD5-Summen (oder SHA-1-Summen) für periodische Überprüfungen erstellt werden.
- Aktualisierte den Abschnitt "Nach der Installation" in Hinblick auf die Konfiguration von checksecurity (war veraltet).

1.6.8 Version 3.3 (Juni 2005)

Änderung von Javier Fernández-Sanguino Peña

- Fügte einen Code-Fetzen hinzu, um mit grep-available eine Liste von Paketen zu erstellen, die von Perl abhängen. Wurde in #302470 nachgefragt.
- Schrieb den Abschnitt über Netzwerkdienste neu (welche installiert sind und wie man sie abschaltet).
- Fügte weitere Informationen zum Abschnitt über die Entwicklung eines Honigtopfs hinzu, indem nützliche Debian-Pakete erwähnt werden.

1.6.9 Version 3.2 (März 2005)

Änderung von Javier Fernández-Sanguino Peña

- Erweiterte den Abschnitt über die Konfiguration von Limits mit PAM.
- Fügte Informationen hinzu, wie pam_chroot für openssh eingesetzt wird (auf Grundlage der README von pam_chroot).
- Verbesserte einige kleinere Dinge, die von Dan Jacobson gemeldet wurden.
- Aktualisierte die Informationen über Kernelpatches, basiert teilweise auf einem Patch von Carlo Perassi, auf den Anmerkungen zu aufgegebenen Teilen des Kernels und auf den neuen Kernelpatches (adamantix).
- Fügte einen Patch von Simon Brandmair ein, der einen Satz im Zusammenhang mit Login-Fehlern auf dem Terminal ausbesserte.
- Fügte Mozilla/Thunderbird zu den gültigen GPG-Agenten hinzu, wie von Kapolnai Richard vorgeschlagen wurde.
- Erweiterte den Abschnitt über Sicherheitsaktualisierungen, die Aktualisierung von Bibliotheken und des Kernels betreffen, und wie man herausfindet, ob Dienste neu gestartet werden müssen.
- Schrieb den Abschnitt über die Firewall neu, habe die Informationen, die Woody betreffen, nach unten verschoben und die übrigen Abschnitte erweitert, einschließlich Hinweisen dazu, wie man von Hand eine Firewall einrichtet (mit einem Beispielsskript) und wie man die Konfiguration der Firewall testen kann.
- Fügte einige Informationen hinzu bezüglich der Veröffentlichung von Debian 3.1.
- Fügte ausführlichere Hinweise zu Kernelupgrades hinzu, die sich besonders an diejenigen richten, die das alte Installationssystem verwenden.

- Fügte einen kurzen Abschnitt über die experimentelle Veröffentlichung von apt 0.6, die die Überprüfung von Paketsignaturen enthält. Verschob den alten Inhalt in den Abschnitt und fügte auch einen Verweis auf die Veränderungen, die in aptitude vorgenommen wurden, hinzu.
- Ausbesserungen von Tippfehlern, die von Frédéric Bothamy entdeckt wurden.

1.6.10 Version 3.1 (January 2005)

Changes by Javier Fernández-Sanguino Peña

- Added clarification to ro /usr with patch from Joost van Baal
- Apply patch from Jens Seidel fixing many typos.
- FreeSWAN is dead, long live OpenSWAN.
- Added information on restricting access to RPC services (when they cannot be disabled) also included patch provided by Aarre Laakso.
- Update aj's apt-check-sigs script.
- Apply patch Carlo Perassi fixing URLs.
- Apply patch from Davor Ocelic fixing many errors, typos, urls, grammar and FIXMEs. Also adds some additional information to some sections.
- Rewrote the section on user auditing, highlight the usage of script which does not have some of the issues associated to shell history.

1.6.11 Version 3.0 (December 2004)

Changes by Javier Fernández-Sanguino Peña

- Rewrote the user-auditing information and include examples on how to use script.

1.6.12 Version 2.99 (March 2004)

Changes by Javier Fernández-Sanguino Peña

- Added information on references in DSAs and CVE-Compatibility.
- Added information on apt 0.6 (apt-secure merge in experimental)
- Fixed location of Chroot daemons HOWTO as suggested by Shuying Wang.

- Changed APACHECTL line in the Apache chroot example (even if its not used at all) as suggested by Leonard Norrgard.
- Added a footnote regarding hardlink attacks if partitions are not setup properly.
- Added some missing steps in order to run bind as named as provided by Jeffrey Prosa.
- Added notes about Nessus and Snort out-of-dateness in woody and availability of backported packages.
- Added a chapter regarding periodic integrity test checks.
- Clarified the status of testing regarding security updates. (Debian bug 233955)
- Added more information regarding expected contents in securetty (since it's kernel specific).
- Added pointer to snoopylogger (Debian bug 179409)
- Added reference to guarddog (Debian bug 170710)
- Apt-ftparchive is in apt-utils, not in apt (thanks to Emmanuel Chantreau for pointing this out)
- Removed jvirus from AV list.

1.6.13 Version 2.98 (December 2003)

Changes by Javier Fernández-Sanguino Peña

- Fixed URL as suggested by Frank Lichtenheld.
- Fixed PermitRootLogin typo as suggested by Stefan Lindenau.

1.6.14 Version 2.97 (September 2003)

Changes by Javier Fernández-Sanguino Peña

- Added those that have made the most significant contributions to this manual (please mail me if you think you should be in the list and are not).
- Added some blurb about FIXME/TODOs
- Moved the information on security updates to the beginning of the section as suggested by Elliott Mitchell.
- Added grsecurity to the list of kernel-patches for security but added a footnote on the current issues with it as suggested by Elliott Mitchell.

- Removed loops (echo to 'all') in the kernel's network security script as suggested by Elliott Mitchell.
- Added more (up-to-date) information in the antivirus section.
- Rewrote the buffer overflow protection section and added more information on patches to the compiler to enable this kind of protection.

1.6.15 Version 2.96 (august 2003)

Changes by Javier Fernández-Sanguino Peña

- Removed (and then re-added) appendix on chrooting Apache. The appendix is now dual-licensed.

1.6.16 Version 2.95 (June 2003)

Changes by Javier Fernández-Sanguino Peña

- Fixed typos spotted by Leonard Norrgard.
- Added a section on how to contact CERT for incident handling (`#after-compromise`)
- More information on setting up a Squid proxy.
- Added a pointer and removed a FIXME thanks to Helge H. F.
- Fixed a typo (save_inactive) spotted by Philippe Faes.
- Fixed several typos spotted by Jaime Robles.

1.6.17 Version 2.94 (April 2003)

Changes by Javier Fernández-Sanguino Peña

- Following Maciej Stachura's suggestions I've expanded the section on limiting users.
- Fixed typo spotted by Wolfgang Nolte.
- Fixed links with patch contributed by Ruben Leote Mendes.
- Added a link to David Wheeler's excellent document on the footnote about counting security vulnerabilities.

1.6.18 Version 2.93 (march 2003)

Changes made by Frédéric Schütz.

- rewrote entirely the section of ext2 attributes (lsattr/chattr)

1.6.19 Version 2.92 (February 2003)

Changes by Javier Fernández-Sanguino Peña and Frédéric Schütz.

- Merge section 9.3 ("useful kernel patches") into section 4.13 ("Adding kernel patches"), and added some content.
- Added a few more TODOs
- Added information on how to manually check for updates and also about cron-apt. That way Tiger is not perceived as the only way to do automatic update checks.
- Slightly rewrite of the section on executing a security updates due to Jean-Marc Ranger comments.
- Added a note on Debian's installation (which will suggest the user to execute a security update right after installation)

1.6.20 Version 2.91 (January/February 2003)

Changes by Javier Fernández-Sanguino Peña (me).

- Added a patch contributed by Frédéric Schütz.
- Added a few more references on capabilities thanks to Frédéric.
- Slight changes in the bind section adding a reference to BIND's 9 online documentation and proper references in the first area (Hi Pedro!)
- Fixed the changelog date - new year :-)
- Added a reference to Colin's articles for the TODOs.
- Removed reference to old ssh+chroot patches.
- More patches from Carlo Perassi.
- Typo fixes (recursive in Bind is recursion), pointed out by Maik Holtkamp.

1.6.21 Version 2.9 (December 2002)

Changes by Javier Fernández-Sanguino Peña (me).

- Reorganized the information on chroot (merged two sections, it didn't make much sense to have them separated)
- Added the notes on chrooting Apache provided by Alexandre Ratti.
- Applied patches contributed by Guillermo Jover.

1.6.22 Version 2.8 (November 2002)

Changes by Javier Fernández-Sanguino Peña (me).

- Applied patches from Carlo Perassi, fixes include: re-wrapping the lines, URL fixes, and fixed some FIXMEs
- Updated the contents of the Debian security team FAQ.
- Added a link to the Debian security team FAQ and the Debian Developer's reference, the duplicated sections might (just might) be removed in the future.
- Fixed the hand-made auditing section with comments from Michal Zielinski.
- Added links to wordlists (contributed by Carlo Perassi)
- Fixed some typos (still many around).
- Fixed TDP links as suggested by John Summerfield.

1.6.23 Version 2.7 (October 2002)

Changes by Javier Fernández-Sanguino Peña (me). Note: I still have a lot of pending changes in my mailbox (which is currently about 5 Mbs in size).

- Some typo fixes contributed by Tuyen Dinh, Bartek Golenko and Daniel K. Gebhart.
- Note regarding /dev/kmem rootkits contributed by Laurent Bonnaud
- Fixed typos and FIXMEs contributed by Carlo Perassi.

1.6.24 Version 2.6 (September 2002)

Changes by Chris Tillman, tillman@voicetrak.com.

- Changed around to improve grammar/spelling.
- s/host.deny/hosts.deny/ (1 place)
- Applied Larry Holish's patch (quite big, fixes a lot of FIXMEs)

1.6.25 Version 2.5 (September 2002)

Changes by Javier Fernández-Sanguino Peña (me).

- Fixed minor typos submitted by Thiemo Nagel.
- Added a footnote suggested by Thiemo Nagel.
- Fixed an URL link.

1.6.26 Version 2.5 (August 2002)

Changes by Javier Fernández-Sanguino Peña (me). There were many things waiting on my inbox (as far back as february) to be included, so I'm going to tag this the *back from honeymoon* release :)

- Added some information on how to setup the Xscreensaver to lock automatically the screen after the configured timeout.
- Add a note related to the utilities you should not install in the system. Including a note regarding Perl and why it cannot be easily removed in Debian. The idea came after reading Intersect's documents regarding Linux hardening.
- Added information on lvm and journaling filesystems, ext3 recommended. The information there might be too generic, however.
- Added a link to the online text version (check).
- Added some more stuff to the information on firewalling the local system triggered by a comment made by Hubert Chan in the mailing list.
- Added more information on PAM limits and pointers to Kurt Seifried's documents (related to a post by him to Bugtraq on April 4th 2002 answering a person that had "discovered" a vulnerability in Debian GNU/Linux related to resource starvation)
- As suggested by Julián Muñoz, provided more information on the default Debian umask and what a user can access if he has been given a shell in the system (scary, huh?)

- Included a note in the BIOS password section due to a comment from from Andreas Wohlfeld.
- Included patches provided by Alfred E. Heggestad fixing many of the typos still present in the document.
- Added a pointer to the changelog in the Credits section since most people who contribute are listed here (and not there)
- Added a few more notes to the chattr section and a new section after installation talking about system snapshots. Both ideas were contributed by Kurt Pomeroy.
- Added a new section after installation just to remember users to change the boot-up sequence.
- Added some more TODO items provided by Korn Andras.
- Added a pointer to the NIST's guidelines on how to secure DNS provided by Daniel Quinlan.
- Added a small paragraph regarding Debian's SSL certificates infrastructure.
- Added Daniel Quinlan's suggestions regarding ssh authentication and exim's relay configuration.
- Added more information regarding securing bind including changes suggested by Daniel Quinlan and an appendix with a script to make some of the changes commented on that section.
- Added a pointer to another item regarding Bind chrooting (needs to be merged)
- Added a one liner contributed by Cristian Ionescu-Idbohrn to retrieve packages with tcpwrappers support.
- Added a little bit more info on Debian's default PAM setup.
- Included a FAQ question about using PAM to give services w/o shell accounts.
- Moved two FAQ items to another section and added a new FAQ regarding attack detection (and compromised systems).
- Included information on how to setup a bridge firewall (including a sample Appendix). Thanks to Francois Bayart who sent me this on march.
- Added a FAQ regarding the syslogd's *MARK heartbeat* from a question answered by Noah Meyerhans and Alain Tesio on December 2001.
- Included information on buffer overflow protection as well as some information on kernel patches.
- Added more information (and reorganised) the firewall section. Updated the information regarding the iptables package and the firewall generators available.

- Reorganized the information regarding logchecking, moved logcheck information from host intrusion detection to that section.
- Added some information on how to prepare a static package for bind for chrooting (untested).
- Added a FAQ item (could be expanded with some of the recomendations from the debian-security list regarding some specific servers/services).
- Added some information on RPC services (and when it's necessary).
- Added some more information on capabilities (and what lcap does). Is there any good documentation on this? I haven't found any on my 2.4 kernel.
- Fixed some typos.

1.6.27 Version 2.4

Changes by Javier Fernández-Sanguino Peña.

- Rewritten part of the BIOS section.

1.6.28 Version 2.3

Changes by Javier Fernández-Sanguino Peña.

- Wrapped most file locations with the file tag.
- Fixed typo noticed by Edi Stojicevi.
- Slightly changed the remote audit tools section.
- Added some todo items.
- Added more information regarding printers and cups config file (taken from a thread on debian-security).
- Added a patch submitted by Jesus Climent regarding access of valid system users to Proftpd when configured as anonymous server.
- Small change on partition schemes for the special case of mail servers.
- Added Hacking Linux Exposed to the books section.
- Fixed directory typo noticed by Eduardo Pérez Ureta.
- Fixed /etc/ssh typo in checklist noticed by Edi Stojicevi.

1.6.29 Version 2.3

Changes by Javier Fernández-Sanguino Peña.

- Fixed location of dpkg conffile.
- Remove Alexander from contact information.
- Added alternate mail address.
- Fixed Alexander mail address (even if commented out).
- Fixed location of release keys (thanks to Pedro Zorzenon for pointing this out).

1.6.30 Version 2.2

Changes by Javier Fernández-Sanguino Peña.

- Fixed typos, thanks to Jamin W. Collins.
- Added a reference to apt-extracttemplate manpage (documents the APT::ExtractTemplate config).
- Added section about restricted SSH. Information based on that posted by Mark Janssen, Christian G. Warden and Emmanuel Lacour on the debian-security mailing list.
- Added information on anti-virus software.
- Added a FAQ: su logs due to the cron running as root.

1.6.31 Version 2.1

Changes by Javier Fernández-Sanguino Peña.

- Changed FIXME from lshell thanks to Oohara Yuuma.
- Added package to sXid and removed comment since it *is* available.
- Fixed a number of typos discovered by Oohara Yuuma.
- ACID is now available in Debian (in the acidlab package) thanks to Oohara Yuuma for noticing.
- Fixed LinuxSecurity links (thanks to Dave Wreski for telling).

1.6.32 Version 2.0

Changes by Javier Fernández-Sanguino Peña. I wanted to change to 2.0 when all the FIXMEs were, er, fixed but I run out of 1.9X numbers :(

- Converted the HOWTO into a Manual (now I can properly say RTFM)
- Added more information regarding tcp wrappers and Debian (now many services are compiled with support for them so it's no longer an inetd issue).
- Clarified the information on disabling services to make it more consistent (rpc info still referred to update-rc.d)
- Added small note on lprng.
- Added some more info on compromised servers (still very rough)
- Fixed typos reported by Mark Bucciarelli.
- Added some more steps in password recovery to cover the cases when the admin has set paranoid-mode=on.
- Added some information to set paranoid-mode=on when login in console.
- New paragraph to introduce service configuration.
- Reorganised the *After installation* section so it is more broken up into several issues and it's easier to read.
- Written information on howto setup firewalls with the standard Debian 3.0 setup (iptables package).
- Small paragraph explaining why installing connected to the Internet is not a good idea and how to avoid this using Debian tools.
- Small paragraph on timely patching referencing to IEEE paper.
- Appendix on how to setup a Debian snort box, based on what Vladimir sent to the debian-security mailing list (September 3rd 2001)
- Information on how logcheck is setup in Debian and how it can be used to setup HIDS.
- Information on user accounting and profile analysis.
- Included apt.conf configuration for read-only /usr copied from Olaf Meeuwissen's post to the debian-security mailing list
- New section on VPN with some pointers and the packages available in Debian (needs content on how to setup the VPNs and Debian-specific issues), based on Jaroslaw Tabor's and Samuli Suonpaa's post to debian-security.
- Small note regarding some programs to automatically build chroot jails

- New FAQ item regarding identd based on a discussion in the debian-security mailing list (February 2002, started by Johannes Weiss).
- New FAQ item regarding inetd based on a discussion in the debian-security mailing list (February 2002).
- Introduced note on rcconf in the "disabling services" section.
- Varied the approach regarding LKM, thanks to Philipe Gaspar
- Added pointers to CERT documents and Counterpane resources

1.6.33 Version 1.99

Changes by Javier Fernández-Sanguino Peña.

- Added a new FAQ item regarding time to fix security vulnerabilities.
- Reorganised FAQ sections.
- Started writing a section regarding firewalling in Debian GNU/Linux (could be broadened a bit)
- Fixed typos sent by Matt Kraai
- Fixed DNS information
- Added information on whisker and nbtscan to the auditing section.
- Fixed some wrong URLs

1.6.34 Version 1.98

Changes by Javier Fernández-Sanguino Peña.

- Added a new section regarding auditing using Debian GNU/Linux.
- Added info regarding finger daemon taken from the security mailing list.

1.6.35 Version 1.97

Changes by Javier Fernández-Sanguino Peña.

- Fixed link for Linux Trustees
- Fixed typos (patches from Oohara Yuuma and Pedro Zorzenon)

1.6.36 Version 1.96

Changes by Javier Fernández-Sanguino Peña.

- Reorganized service installation and removal and added some new notes.
- Added some notes regarding using integrity checkers as intrusion detection tools.
- Added a chapter regarding package signatures.

1.6.37 Version 1.95

Changes by Javier Fernández-Sanguino Peña.

- Added notes regarding Squid security sent by Philipe Gaspar.
- Fixed rootkit links thanks to Philipe Gaspar.

1.6.38 Version 1.94

Changes by Javier Fernández-Sanguino Peña.

- Added some notes regarding Apache and Lpr/lpng.
- Added some information regarding noexec and read-only partitions.
- Rewritten how can users help in Debian security issues (FAQ item).

1.6.39 Version 1.93

Changes by Javier Fernández-Sanguino Peña.

- Fixed location of mail program.
- Added some new items to the FAQ.

1.6.40 Version 1.92

Changes by Javier Fernández-Sanguino Peña.

- Added a small section on how Debian handles security
- Clarified MD5 passwords (thanks to 'rocky')
- Added some more information regarding harden-X from Stephen van Egmond
- Added some new items to the FAQ

1.6.41 Version 1.91

Changes by Javier Fernández-Sanguino Peña.

- Added some forensics information sent by Yotam Rubin.
- Added information on how to build a honeynet using Debian GNU/Linux.
- Added some more TODOS.
- Fixed more typos (thanks Yotam!)

1.6.42 Version 1.9

Changes by Javier Fernández-Sanguino Peña.

- Added patch to fix misspellings and some new information (contributed by Yotam Rubin)
- Added references to other online (and offline) documentation both in a section (see 'Seien Sie wachsam gegenüber generellen Sicherheitsproblemen!' auf Seite 29) by itself and inline in some sections.
- Added some information on configuring Bind options to restrict access to the DNS server.
- Added information on how to automatically harden a Debian system (regarding the harden package and bastille).
- Removed some done TODOs and added some new ones.

1.6.43 Version 1.8

Changes by Javier Fernández-Sanguino Peña.

- Added the default user/group list provided by Joey Hess to the debian-security mailing list.
- Added information on LKM root-kits ('Ladbare Kernel-Module (LKM)' auf Seite 189) contributed by Philipe Gaspar.
- Added information on Proftp contributed by Emmanuel Lacour.
- Recovered the checklist Appendix from Era Eriksson.
- Added some new TODO items and removed other fixed ones.
- Manually included Era's patches since they were not all included in the previous version.

1.6.44 Version 1.7

Changes by Era Eriksson.

- Typo fixes and wording changes

Changes by Javier Fernández-Sanguino Peña.

- Minor changes to tags in order to keep on removing the tt tags and substitute them for prgn/package tags.

1.6.45 Version 1.6

Changes by Javier Fernández-Sanguino Peña.

- Added pointer to document as published in the DDP (should supersede the original in the near future)
- Started a mini-FAQ (should be expanded) with some questions recovered from my mailbox.
- Added general information to consider while securing.
- Added a paragraph regarding local (incoming) mail delivery.
- Added some pointers to more information.
- Added information regarding the printing service.
- Added a security hardening checklist.
- Reorganized NIS and RPC information.
- Added some notes taken while reading this document on my new Visor :)
- Fixed some badly formatted lines.
- Fixed some typos.
- Added a Genius/Paranoia idea contributed by Gaby Schilders.

1.6.46 Version 1.5

Changes by Josip Rodin and Javier Fernández-Sanguino Peña.

- Added paragraphs related to BIND and some FIXMEs.

1.6.47 Version 1.4

- Small setuid check paragraph
- Various minor cleanups
- Found out how to use `sgml2txt -f` for the txt version

1.6.48 Version 1.3

- Added a security update after installation paragraph
- Added a proftpd paragraph
- This time really wrote something about XDM, sorry for last time

1.6.49 Version 1.2

- Lots of grammar corrections by James Treacy, new XDM paragraph

1.6.50 Version 1.1

- Typo fixes, miscellaneous additions

1.6.51 Version 1.0

- Initial release

1.7 Danksagungen

- Alexander Reelsen schrieb die ursprüngliche Version.
- Javier Fernández-Sanguino fügte der Originalversion einiges an Informationen hinzu.
- Robert van der Meulen stellte den Abschnitt über Quota und viele seiner guten Ideen zur Verfügung.
- Ethan Benson korrigierte den PAM-Abschnitt und hatte einige gute Ideen.
- Dariusz Puchalak trug Information zu verschiedenen Kapiteln bei.
- Gaby Schilders trug eine nette Genius/Paranoia Idee bei.
- Era Eriksson gab dem Ganzen an vielen Stellen den sprachlichen Feinschliff und trug zur Checkliste im Anhang bei.

- Philipe Gaspar schrieb die LKM-Informationen.
- Yotam Rubin trug sowohl Korrekturen für viele Tippfehler bei als auch Informationen über die Versionen von Bind und MD5-Passwörter.
- Francois Bayart stellte den Anhang zur Verfügung, in dem beschrieben wird, wie man eine Bridge-Firewall aufsetzt.
- Joey Hess schrieb im Debian-Wiki (`wiki.debian.org`) den Abschnitt, der erklärt, wie Secure Apt funktioniert.
- Martin F. Krafft schrieb in seinem Blog etwas darüber, wie die Verifizierung von Fingerprints funktioniert. Dies wurde im Abschnitt über Secure Apt verwendet.
- All den Leuten, die Verbesserungen vorschlugen, die (letzten Endes) eingeflossen sind (siehe 'Änderungsübersicht/Changelog/Geschichte' auf Seite 7).
- (Alexander) All den Leuten, die mich ermutigten dieses HOWTO zu schreiben (die später zu einer ganzen Anleitung wurde).
- Dem ganzen Debian-Projekt.

1.7.1 Danksagungen des Übersetzers

Auch der Übersetzer Simon Brandmair (`<sbrandmair@gmx.net>`) hat einigen Leuten zu danken, die mit Verbesserungen, Korrekturen und Ratschlägen zum Gelingen der Übersetzung beigetragen haben:

- Christoph Haas
- Mirko Jahn
- Frank Loeffler
- Andreas Marc Klingler
- Elisabeth Bauer
- Jens Kubieziel
- Uli Martens
- Jens Schuessler
- Marcel Schaal
- Jens Seidel
- Constantin Hagemeier

Besonders möchte ich Alexander Schmehl danken, der die erste deutsche Übersetzung angefertigt hat.

Insbesondere sei den Mitglieder der debian-l10n-german (http://lists.debian.org/debian-l10n-german/) Mailingliste gedankt, für gute Ideen und fruchtbare Diskussionen.

VIELEN DANK! Ohne euch hätte ich zwar nur halb so viel Arbeit gehabt, aber viel mehr Leute könnten sich jetzt über meine mangelhaften Orthographiekenntnisse amüsieren!

Kapitel 2

Bevor Sie beginnen ...

2.1 Wofür möchten Sie dieses System benutzen?

Das Absichern von Debian ist nicht viel anders als das Absichern von irgendeinem anderen System. Um es richtig zu machen müssen Sie zunächst entscheiden, was Sie damit machen möchten. Anschließend müssen Sie sich klarmachen, dass Sie die folgenden Schritte sorgfältig ausgeführt werden müssen, um ein wirklich sicheres System zu bekommen.

Sie werden feststellen, dass diese Anleitung von der Pike auf geschrieben ist. Sie werden die Informationen zu einer Aufgabe, die Sie vor, während und nach der Debian-Installation ausführen sollten, in der entsprechenden Reihenfolge vorgestellt bekommen. Die einzelnen Aufgaben können wie folgt beschrieben werden:

- Entscheiden Sie, welche Dienste Sie benötigen, und beschränken Sie Ihr System auf selbige. Dies schließt das Deaktivieren / Deinstallieren von nicht benötigten Diensten und das Installieren von Firewall-ähnlichen Filtern oder TCP-Wrappern ein.
- Einschränken der Nutzer- und Zugriffsrechte auf Ihrem System.
- Abhärten der angebotenen Dienste, damit der Einfluss auf Ihr System im Falle einer Kompromittierung möglichst gering ist.
- Benutzen Sie die passenden Tools, um sicherzustellen, dass ein unautorisierter Zugriff auf Ihrem System entdeckt wird, so dass Sie geeignete Gegenmaßnahmen ergreifen können.

2.2 Seien Sie wachsam gegenüber generellen Sicherheitsproblemen!

Diese Anleitung geht (normalerweise) nicht im Detail darauf ein, warum bestimmte Sachen als Sicherheitsrisiko betrachtet werden. Es wäre aber sicherlich von Vorteil, wenn Sie mehr Hintergrundwissen von der Sicherheit in Unix im Allgemeinen und von der in Linux im Besonderen haben. Nehmen Sie sich die Zeit, um sicherheitsrelevante Dokumente zu lesen, um

Entscheidungen informiert treffen zu können, wenn Sie eine Auswahl treffen müssen. Debian GNU/Linux basiert auf dem Linux-Kernel, so dass viele Informationen über Linux, und sogar über andere Distributionen und allgemeine UNIX-Sicherheit, auch hierauf zutreffen (sogar wenn sich die benutzten Werkzeuge oder die verfügbaren Programme unterscheiden).

Ein Paar nützliche Dokumente sind:

- Das Linux Security HOWTO (http://www.tldp.org/HOWTO/Security-HOWTO/) (auch unterLinuxSecurity (http://www.linuxsecurity.com/docs/LDP/Security-HOWTO.html) verfügbar) ist eine der besten Referenzen über allgemeine Linux-Sicherheit.
- Das Security Quick-Start HOWTO for Linux (http://www.tldp.org/HOWTO/Security-Quickstart-HOWTO/) ist ein sehr guter Anfang für unerfahrene Nutzer (sowohl über Linux als auch zum Thema Sicherheit).
- Der Linux Security Administrator's Guide (http://seifried.org/lasg/) ist eine komplette Anleitung, die alle Sicherheitsangelegenheiten von Linux behandelt, von Sicherheit im Kernel bis hin zu VPNs. Beachten Sie bitte, dass er seit 2001 nicht mehr aktualisiert wurde, trotzdem sind einige Informationen immer noch sachdienlich. [1]
- Kurt Seifried's Securing Linux Step by Step (http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html).
- In Securing and Optimizing Linux: RedHat Edition (http://www.tldp.org/links/p_books.html#securing_linux) finden Sie eine Dokumentation ähnlich zu dieser, bezogen auf Red Hat. Manche behandelten Sachen sind nicht distributionsspezifisch, passen also auch auf Debian.
- Ein anderes Red Hat bezogenes Dokument ist der EAL3 Evaluated Configuration Guide for Red Hat Enterprise (http://ltp.sourceforge.net/docs/RHEL-EAL3-Configuration-Guide.pdf).
- IntersectAlliance hat einige Dokumente veröffentlicht, die als Referenz benutzt werden können, wie man einen Linux-Server (und seine Dienste) abhärtet. Diese Dokumente sind auf ihrer Seite (http://www.intersectalliance.com/projects/index.html) verfügbar.
- Für Netzwerk-Administratoren ist das Securing your Domain HOWTO (http://www.linuxsecurity.com/docs/LDP/Securing-Domain-HOWTO/) ein gutes Handbuch, wie man sein Netzwerk absichert.
- Wenn Sie die Programme, die Sie benutzen möchten (oder die Sie neu schreiben wollen), bezüglich Sicherheit auswerten wollen, sollten Sie das Secure Programs HOWTO (http://www.tldp.org/HOWTO/Secure-Programs-HOWTO/) durchlesen (die Vorlage ist unter http://www.dwheeler.com/secure-programs/ verfügbar. Es beinhaltet Präsentationen und Kommentare des Autors David Wheeler.

[1] Irgendwann wurde er von der "Linux Security Knowledge Base" abgelöst. Dieses Dokument wird ebenfalls durch das Paket `lasg` zur Verfügung gestellt. Jetzt wird der Guide wieder unter dem Namen *Lasg* verbreitet.

- Wenn Sie es in Betracht ziehen, eine Firewall zu installieren, sollten Sie das Firewall HOWTO (http://www.tldp.org/HOWTO/Firewall-HOWTO.html) und das IPCHAINS HOWTO (http://www.tldp.org/HOWTO/IPCHAINS-HOWTO.html) (bei Kerneln vor Version 2.4) lesen.
- Schließlich ist die Linux Security ReferenceCard (http://www.linuxsecurity.com/docs/QuickRefCard.pdf) eine gute Kurzübersicht, um in Sachen Sicherheit auf dem aktuellen Stand zu bleiben.

In jedem Fall gibt es mehr Informationen über die hier behandelten Dienste (NFS, NIS, SMB, . . .) in den vielen HOWTOs, die Sie beim Linux-Dokumentations-Projekt (http://www.tldp.org/) finden. Manche dieser Dokumente gehen auf die Sicherheitsaspekte von bestimmten Diensten ein. Gehen Sie sicher, dass Sie auch hierauf einen Blick werfen.

Die HOWTO-Dokumente des Linux-Dokumentations-Projektes sind unter Debian GNU/Linux durch Installation der Pakete `doc-linux-text` (englische Text-Version) oder `doc-linux-de` (HTML-Version) verfügbar. Nach der Installation sind diese Dokumente in den Verzeichnissen `/usr/share/doc/HOWTO/en-txt` beziehungsweise `/usr/share/doc/HOWTO/de-html` vorhanden.

Andere empfohlene Linux-Bücher:

- Maximum Linux Security : A Hacker's Guide to Protecting Your Linux Server and Network. Anonymous. Paperback - 829 pages. Sams Publishing. ISBN: 0672313413. July 1999.
- Linux Security By John S. Flowers. New Riders; ISBN: 0735700354. March 1999
- Hacking Linux Exposed (http://www.linux.org/books/ISBN_0072127732.html) By Brian Hatch. McGraw-Hill Higher Education. ISBN 0072127732. April, 2001

Andere Bücher (auch über allgemeine Aspekte von Sicherheit unter Unix, nicht nur Linuxspezifisch):

- Practical Unix and Internet Security (2nd Edition) (http://www.ora.com/catalog/puis/noframes.html) Garfinkel, Simpson, and Spafford, Gene; O'Reilly Associates; ISBN 0-56592-148-8; 1004pp; 1996.
- Firewalls and Internet Security Cheswick, William R. and Bellovin, Steven M.; Addison-Wesley; 1994; ISBN 0-201-63357-4; 320pp.

Andere nützliche Webseiten, um sich bezüglich Sicherheit auf dem Laufenden zu halten:

- NIST Security Guidelines (http://csrc.nist.gov/fasp/index.html).
- Security Focus (http://www.securityfocus.com) Dort wird die Bugtraq-Schwachstellen-Datenbank und Mailingliste bereitgestellt und es gibt allgemeine sicherheitsrelevante Informationen, Neuigkeiten und Berichte.

- Linux Security (http://www.linuxsecurity.com/). Allgemeine Informationen zu Sicherheit von Linux (Tools, Neuigkeiten ...). Die Seite main documentation (http://www.linuxsecurity.com/resources/documentation-1.html) ist sehr nützlich.
- Linux firewall and security site (http://www.linux-firewall-tools.com/linux/). Allgemeine Informationen zu Linux Firewalls und Tools, diese zu kontrollieren und zu administrieren.

2.3 Wie geht Debian mit der Sicherheit um?

Um einen allgemeinen Überblick über die Sicherheit unter Debian GNU/Linux zu bekommen, sollten Sie sich ansehen, was Debian tut, um ein sicheres System zu gewährleisten.

- Debians Probleme werden immer öffentlich behandelt, sogar wenn sie die Sicherheit betreffen. Sicherheitsfragen werden öffentlich auf der debian-security-Mailingliste diskutiert. Debian-Sicherheits-Ankündigungen (DSA) werden an öffentliche Mailinglisten (sowohl intern als auch extern) versendet und auf öffentlichen Servern bekannt gegeben. Wie der Debian-Gesellschaftsvertrag (http://www.debian.org/social_contract) sagt:

  Wir werden Probleme nicht verbergen.

  Wir werden unsere Fehlerdatenbank immer öffentlich betreiben. Fehlermeldungen, die von Personen online abgeschickt werden, sind augenblicklich für andere sichtbar.

- Debian verfolgt Sicherheitsangelegenheiten sehr aufmerksam. Das Sicherheits-Team prüft viele sicherheitsrelevante Quellen, die wichtigste davon Bugtraq (http://www.securityfocus.com/cgi-bin/vulns.pl), während es Pakete mit Sicherheitsproblemen sucht, die ein Teil von Debian sein können.
- Sicherheits-Aktualisierungen genießen höchste Priorität. Wenn ein Sicherheitsproblem in einem Debian-Paket entdeckt wird, wird eine Sicherheits-Aktualisierung so schnell wie möglich vorbereitet und für den Stabile-, Testing- und Unstabile-Zweig, einschließlich aller Architekturen, veröffentlicht.
- Alle Informationen über Sicherheit sind an einer zentralen Stelle zu finden: http://security.debian.org/.
- Debian versucht immer, die gesamte Sicherheit seiner Distribution zu verbessern, beispielsweise durch automatische Paket-Signierungs- und Verifikations-Mechanismen.
- Debian stellt eine brauchbare Anzahl von sicherheitsrelevanten Werkzeugen für System-Administratoren und zur Überwachung zur Verfügung. Entwickler versuchen, diese Werkzeuge fest mit der Distribution zu verbinden, um Sie anpassbarer zur Durchsetzung lokaler Sicherheits-Regelungen zu machen. Diese Werkzeuge schließen Folgendes mit ein: integritätsprüfende Programme, allgemeine Prüfwerkzeuge, Werkzeuge zum Abhärten, Werkzeuge für Firewalls, Eindringlings-Erkennungs-Tools und vieles andere.

- Paketbetreuer sind sich der Sicherheits-Probleme bewusst. Dies führt oft zu “voreingestellt sicheren” Installationen von Diensten, die sie manchmal in ihrer normalen Benutzung etwas einschränken können. Dennoch versucht Debian, Sicherheitsaspekte und Einfachheit der Administration abzuwägen, zum Beispiel werden Dienste nicht inaktiv installiert (wie es bei den Betriebssystemen der BSD-Familie üblich ist). Auf jeden Fall sind bedeutende Sicherheitsaspekte, wie zum Beispiel `setuid`-Programme, Teil der Debian Policy (`http://www.debian.org/doc/debian-policy/`).

Dieses Dokument versucht, eine bessere Installation von Computersystemen hinsichtlich der Sicherheit zu erzielen, indem es Informationen über Sicherheit veröffentlicht, die auf Debian zugeschnitten sind, und diese durch andere Dokumente ergänzt, die sicherheitsspezifische Angelegenheiten im Zusammenhang mit Debian behandeln (vergleiche ‘Seien Sie wachsam gegenüber generellen Sicherheitsproblemen!’ auf Seite 29).

Kapitel 3

Vor und während der Installation

3.1 Setzen Sie ein Passwort im BIOS

Bevor Sie irgendein Betriebssystem auf Ihrem Computer installieren, setzen Sie ein Passwort im BIOS. Nach der Installation (sobald Sie von der Festplatte booten können) sollten Sie zurück ins BIOS gehen und die Boot-Reihenfolge ändern, so dass Sie nicht von Diskette, CD-ROM oder sonstigen Geräten booten können, von denen dies nicht gehen sollte. Andernfalls benötigt ein Cracker nur physischen Zugang und eine Bootdiskette, um Zugriff auf Ihr ganzes System zu bekommen.

Es ist noch besser, wenn das System beim Booten immer ein Passwort verlangt. Dies kann sehr effektiv sein, wenn Sie einen Server laufen lassen, der selten neu gestartet wird. Der Nachteil dieser Vorgehensweise ist, dass das Neustarten einen menschlichen Eingriff benötigt, was zu Problemen führen kann, wenn das System nicht leicht zugänglich ist.

Beachten Sie: Viele BIOS-Varianten haben bekannte Master-Passwörter und es gibt sogar Programme, um Passwörter aus dem BIOS wieder auszulesen. Folglich können Sie sich nicht auf diese Maßnahme verlassen, um den Zugriff auf das Systems zu beschränken.

3.2 Partitionieren des Systems

3.2.1 Wählen Sie eine sinnvolle Partitionierung

Was eine sinnvolle Partitionierung ist, hängt davon ab, wie die Maschine benutzt wird. Eine gute Faustregel ist, mit Ihren Partitionen eher großzügig zu sein und die folgenden Faktoren zu berücksichtigen:

- Jeder Verzeichnisbaum, auf den ein Nutzer Schreibzugriff hat (wie zum Beispiel `/home`, `/tmp` und `/var/tmp/`) sollte auf einer separaten Partition liegen. Dies reduziert das Risiko eines DoS durch einen Nutzer, indem er Ihren "/"-Mountpoint vollschreibt und so das

gesamte System unbenutzbar macht. [1] Außerdem verhindert dieses Vorgehen Hardlink-Angriffe. [2]

- Außerdem sollte jeder Verzeichnisbaum, dessen Größe schwanken kann, zum Beispiel `/var` (insbesondere `/var/log`) eine separate Partition bekommen. Auf einem Debian-System sollten Sie der /var-Partition etwas mehr Platz als auf anderen Systemen geben, da heruntergeladene Pakete (der Zwischenspeicher von apt) unter `/var/cache/apt/archives` gespeichert werden.
- Jeder Teil, in dem Sie Nicht-Distributions-Software installieren wollen, sollte eine separate Partition erhalten. Nach dem File-Hierarchy-Standard wären dies `/opt` oder `/usr/local`. Wenn dies separate Partitionen sind, werden sie nicht gelöscht, falls Sie einmal Ihr Debian neu installieren (müssen).
- Rein sicherheitstechnisch ist es sinnvoll zu versuchen, statische Daten auf eine eigene Partition zu legen und diese dann als nur-lesbar einzuhängen (mounten). Oder noch besser: legen Sie diese Daten auf einem nicht beschreibbares Medium ab. Lesen Sie dazu die Ausführungen weiter unten.

Im Falle eines Mailservers ist es wichtig, eine separate Partition für die Mail-Warteschlange (mail spool) anzulegen. Nicht-Lokale Nutzer können (wissentlich oder unwissentlich) diese Verzeichnisse (`/var/mail` oder `/var/spool/mail`) überfüllen. Liegt dieses Verzeichnis auf einer separaten Partition, würde dies das System nicht sofort unbenutzbar machen. Anderenfalls (wenn das Verzeichnis auch auf der `/var`-Partition liegt) hat das System ein großes Problem: Protokoll-Einträge (logs) können nicht erstellt werden, Pakete können nicht installiert werden und es könnten sogar ein paar Programme Probleme mit dem Starten haben (wenn Sie `/var/run` benutzen).

Außerdem sollten Sie für Partitionen, deren Platzbedarf Sie noch nicht abschätzen können, den Logical-Volume-Manager (`lvm-common` und die benötigten ausführbaren Programme, entweder `lvm10` oder `lvm2`) installieren. Durch Benutzen von `lvm` können Sie Datenträger-Gruppen erstellen, die über mehrere Festplatten verteilt sind.

Auswahl eines passenden Dateisystems

Während der Partitionierung des Systems müssen Sie sich ebenfalls entscheiden, welche Dateisysteme Sie benutzen möchten. Als Standard-Dateisystem wird während der Installa-

[1] Eigentlich ist das so nicht ganz richtig, da immer etwas Platz für Root reserviert wird, den ein normaler Nutzer nicht belegen kann.

[2] Ein sehr gutes Beispiel dieser Art von Angriff, der das /tmp-Verzeichnis benutzt, ist ausführlich auf The mysteriously persistently exploitable program (contest) (`http://www.hackinglinuxexposed.com/articles/20031111.html`) und auf The mysteriously persistently exploitable program explained (`http://www.hackinglinuxexposed.com/articles/20031214.html`) beschrieben (beachten Sie, dass dieser Vorfall in einem Zusammenhang mit Debian steht). Im Prinzip ist das ein Angriff, bei dem ein lokaler Benutzer eine angreifbare Setuid-Anwendung *versteckt*, indem einer einen harten Link zu ihr einrichtet. So kann er wirksam verhindern, dass diese Anwendung vom Systemadministrator aktualisiert (oder entfernt) wird. Dpkg wurde kürzlich verbessert, um das zu verhindern (vergleiche 225692 (`http://bugs.debian.org/225692`)). Aber andere Setuid-Anwendungen, die nicht vom Paketverwaltungsprogramm kontrolliert werden, bleiben ein Risiko, wenn Partitionen nicht richtig eingerichtet werden.

tion für Linux Partitionen `ext2` ausgewählt. Dennoch ist es ratsam, ein "journaling filesystem" (ein Dateisystem, das Änderungen mitprotokolliert) zu nehmen, wie zum Beispiel `ext3`, `reiserfs`, `jfs` oder `xfs`. Dadurch verringern Sie Probleme nach einen Absturz des Systems in folgenden Fällen:

- Auf Laptops auf allen Dateisystemen. Auf diese Art reduzieren Sie die Wahrscheinlichkeit eines Datenverlustes, wenn beispielsweise unerwartet Ihr Akku leer wird oder das System aufgrund eines Hardware-Problems (etwa durch die X-Konfiguration, was relativ häufig auftritt) neu gestartet werden muss.
- Auf produktiven Systemen, die große Mengen von Daten speichern (zum Beispiel Mail-Server, FTP-Server, Netzwerk-Fileserver, ...), ist es empfehlenswert, ein Journaling-Dateisystem auf diesen Partitionen einzusetzen. Wenn das System abstürzt, benötigt der Server so weniger Zeit, um das Dateisystem wieder herzustellen und zu prüfen, und die Wahrscheinlichkeit eines Datenverlustes wird verringert.

Lassen wir mal die Betrachtung der Leistung von Journaling-Dateisystemen beiseite (da dies oft in quasi-religiöse Glaubenskriege ausartet). In der Regel ist es besser, das `ext3`-Dateisystem zu benutzen. Der Grund dafür ist die Abwärtskompatibilität zu `ext2`. So können Sie, wenn es Probleme mit dem Journal gibt, dieses einfach abschalten und haben immer noch ein funktionierendes Dateisystem. Außerdem müssen Sie, wenn Sie das System mal mit einer Boot-Diskette (oder CD-ROM) wiederherstellen müssen, keinen speziellen Kernel benutzen. Wenn es sich um einen 2.4er oder 2.6er Kernel handelt, ist Unterstützung für `ext3` bereits vorhanden. Wenn es sich um einen 2.2er-Kernel handelt, können Sie trotzdem Ihr Dateisystem booten, auch wenn Sie die Journaling-Fähigkeiten einbüßen. Wenn Sie ein anderes Journaling-Dateisystem benutzen, werden Sie feststellen, dass eine Wiederherstellung nicht möglich ist, bis Sie einen 2.4er oder 2.6er Kernel mit den benötigten Modulen haben. Wenn Sie einen 2.2er Kernel auf der Rettungsdiskette verwenden müssen, kann es sich als noch schwerer erweisen, auf `reiserfs` oder `xfs` zuzugreifen.

Auf jeden Fall ist die Datenintegrität unter `ext3` besser, da es auch Datei-Daten protokolliert, während andere Dateisysteme lediglich Meta-Daten protokollieren (siehe auch `http://lwn.net/2001/0802/a/ext3-modes.php3`).

3.3 Gehen Sie nicht ins Internet, bevor Sie nicht bereit sind

Während der Installation sollten Sie das System nicht sofort mit dem Internet verbinden. Dies hört sich vielleicht komisch an, aber die Installation über das Netzwerk ist eine gängige Methode. Da das System einige Dienste installiert und diese sofort aktiviert werden, könnten Sie Ihr System für Angriffe öffnen, wenn das System mit dem Internet verbunden ist und die Dienste nicht geeignet konfiguriert sind.

Außerdem sollten Sie beachten, dass manche Pakete noch Sicherheitsprobleme haben können, weil das Installationsmedium nicht auf dem aktuellen Stand ist. Dies ist für gewöhnlich dann der Fall, wenn Sie von älteren Medien (wie CD-ROMs) installieren. In diesem Fall könnte Ihr System bereits kompromittiert sein, bevor Sie mit der Installation fertig sind!

Da die Debian-Installation und das Aktualisieren über das Internet durchgeführt werden können, denken Sie vielleicht, es sei eine gute Idee, dies gleich während der Installation zu nutzen. Wenn das System direkt mit dem Internet verbunden ist (und nicht von einer Firewall oder NAT geschützt wird), ist es besser, das System ohne Internet-Verbindung zu installieren. Benutzen Sie sowohl für die zu installierenden Pakete als auch für die Sicherheits-Updates eine lokale Quelle (mirror). Sie können einen Paket-Mirror aufsetzen, indem Sie ein anderes System nutzen, dass mit dem Internet verbunden ist und für Debian spezifische Werkzeuge (falls es sich um ein Debian-System handelt) wie `apt-move` oder `apt-proxy` oder andere gebräuchliche Werkzeuge zur Erstellung von Quellen verwendet. Damit kann das Archiv für das installierte System zur Verfügung gestellt werden. Sollte dies nicht möglich sein, sollten Sie Firewall-Regeln aufsetzen, die den Zugriff auf Ihr System beschränken, während Sie das Update durchführen (siehe 'Schutz der Sicherheitsaktualisierung durch eine Firewall' auf Seite 247).

3.4 Setzen Sie ein Passwort für root

Die wichtigste Grundlage für ein sicheres System ist ein gutes Root-Passwort. Siehe `passwd(1)` für einige Tipps, wie man gute Passwörter auswählt. Sie können auch automatische Passwort-Generatoren verwenden (siehe 'Erstellen von Benutzerpasswörtern' auf Seite 71).

Im Internet gibt es zahlreiche Hinweise dazu, wie man gute Passwörter wählt. Zwei Seiten, die eine angemessene Übersicht und Ausführung bieten, sind Eric Wolframs How to: Pick a Safe Password (`http://wolfram.org/writing/howto/password.html`) und Walter Belgers Unix Password Security (`http://www.ja.net/CERT/Belgers/UNIX-password-security.html`).

3.5 Aktivieren Sie Shadow-Passwörter und MD5-Passwörter

Gegen Ende der Installation werden Sie gefragt, ob "shadow passwords" eingeschaltet werden sollen. Beantworten Sie diese Frage mit "yes", dann werden Passwörter in der Datei `/etc/shadow` gespeichert. Nur root und die Gruppe shadow haben Lesezugriff auf diese Datei. So ist es keinem Nutzer möglich, sich eine Kopie dieser Datei zu besorgen, um einen Passwort-Cracker auf sie loszulassen. Sie können mit dem Befehl `shadowconfig` jederzeit zwischen "shadow passwords" und normalen Passwörtern wechseln.

Mehr zum Thema Shadow-Passwörter finden Sie unter Shadow Password (`http://www.tldp.org/HOWTO/Shadow-Password-HOWTO.html`) (`/usr/share/doc/HOWTO/en-txt/Shadow-Password.txt.gz`).

Des Weiteren verwendet die Installation standardmäßig Passwörter, welche als MD5-Hashwerte gespeichert werden. Dies ist im Allgemeinen eine sehr gute Idee, da es längere Passwörter und bessere Verschlüsselung erlaubt. Mit MD5 sind Passwörter möglich, die länger als 8 Zeichen sind. Auf diese Weise kann man es einem Angreifer schwieriger machen, mit

Brute-Force-Methoden an die Passwörter heranzukommen. Dies ist die Standardeinstellung in den neuesten Versionen des Pakets `passwd`. Sie erkennen MD5-Passwörter übrigens in der `/etc/shadow` an dem Anfang 1.

Dies verändert allerdings alle Dateien im Verzeichnis `/etc/pam.d` und ergänzt in der "password"-Zeile den Eintrag "md5".

```
password required pam_unix.so md5 nullok obscure min=6 max=16
```

Wird für `max` nicht ein Wert größer als 8 gewählt, ist diese Änderung ziemlich sinnlos. Weitere Informationen dazu finden Sie unter 'Nutzerauthentifizierung: PAM' auf Seite 58.

Beachten Sie: Die Standardeinstellung in Debian verändert nicht den vorher gewählten `max`-Wert, auch dann nicht, wenn MD5 aktiviert wird.

3.6 Lassen Sie so wenige Dienste wie möglich laufen

Dienste sind Programme wie FTP- und Web-Server. Da sie nach eingehenden Verbindungen, die den Dienst anfordern, *horchen* müssen, können sich externe Computer mit Ihrem Computer verbinden. Dienste sind manchmal verwundbar (zum Beispiel durch einen bestimmten Angriff kompromittierbar) und stellen dadurch ein Sicherheitsrisiko dar.

Sie sollten keine Dienste installieren, die Sie nicht unbedingt auf dem System brauchen. Jeder installierte Dienst könnte neue, vielleicht nicht gerade offensichtliche (oder bekannte) Sicherheitslöcher auf Ihrem Computer öffnen.

Wie Sie vielleicht schon wissen, wird ein Dienst, sobald er installiert wird, auch gleich automatisch aktiviert. Bei einer Standardinstallation ohne weitere installierte Dienste ist die Anzahl der laufenden Dienste ziemlich gering. Und die Anzahl der Dienste, die im Netzwerk angeboten werden, ist noch niedriger. In einer Standardinstallation von Debian 3.1 werden Sie mit OpenSSH, Exim (abhängig davon, wie Sie ihn konfiguriert haben) und dem RPC-Portmapper als Netzwerkdienste auskommen.[3] Der RPC-Portmapper ist standardmäßig installiert, da er für viele Dienste wie zum Beispiel NFS benötigt wird. Er kann allerdings sehr leicht entfernt werden. Weitere Informationen, wie Sie RPC-Dienste absichern oder abschalten, finden Sie unter 'Sichern von RPC-Diensten' auf Seite 120.

Wenn Sie einen neuen Netzwerkdienst (Daemon) auf Ihrem Debian GNU/Linux System installieren, kann er auf zwei Arten gestartet werden: durch den `inetd`-Superdaemon (d. h. eine Zeile wird zu der `/etc/inetd.conf` hinzugefügt) oder durch ein eigenständiges Programm, dass sich selbst an die Netzwerkschnittstelle bindet. Eigenständige Programme werden durch `/etc/init.d` gesteuert. Sie werden beim Hochfahren durch den Sys-V-Mechanismus gestartet, der die symbolischen Links in `/etc/rc?.d/*` benutzt. (Mehr Informationen dazu finden Sie in `/usr/share/doc/sysvinit/README.runlevels.gz`).

[3] Die Zahl war bei Debian 3.0 und davor nicht so niedrig, da einige `inetd`-Dienste standardmäßig aktiviert waren. Außerdem war in Debian 2.2 der NFS-Server wie auch der Telnet-Server Bestandteil der Standardinstallation.

Wenn Sie trotzdem Dienste installieren möchten, diese aber selten benutzen, entfernen Sie sie mit den update-Befehlen wie `update-inetd` oder `update-rc.d` aus dem Startvorgang. Weitere Informationen, wie Sie Netzwerkdienste abschalten, finden Sie unter 'Daemons abschalten' auf dieser Seite. Wenn Sie das Standardverhalten des Startens von Diensten nach der Installation von ihren Paketen ändern wollen[4], lesen Sie bitte für weiterführende Informationen `/usr/share/doc/sysv-rc/README.policy-rc.d.gz`.

Die Unterstützung von `invoke-rc.d` ist bei Debian nun zwingend. Dies bedeutet, dass Sie bei *Sid* und *Etch* eine `policy-rc.d`-Datei anlegen können, die das Starten von Daemons verbietet, bevor Sie sie konfiguriert haben. Zwar sind derartige Skripte noch nicht in Paketen enthalten, sie sind aber ziemlich leicht zu schreiben. Sehen Sie sich auch `policyrcd-script-zg2` an.

3.6.1 Daemons abschalten

Das Abschalten eines Daemons ist sehr einfach. Entweder Sie entfernen das Paket, das das Programm für diesen Dienst anbietet, oder Sie entfernen oder benennen die Startlinks unter `/etc/rc${runlevel}.d/` um. Wenn Sie sie umbenennen, stellen Sie sicher, dass sie nicht mehr mit einem 'S' beginnen, damit sie nicht von `/etc/init.d/rc` ausgeführt werden. Entfernen Sie nicht alle verfügbaren Links, denn sonst wird das Paketverwaltungssystem sie bei der nächsten Paketaktualisierung wieder herstellen. Gehen Sie also sicher, dass zumindest ein Link übrig bleibt (typischerweise ein 'K'-Link, 'K' steht für 'kill'). Zusätzliche Informationen finden Sie im Abschnitt Anpassen des Bootvorgangs (`http://www.debian.org/doc/manuals/reference/ch-system#s-custombootscripts`) der Debian-Referenz (2. Kapitel - Debian-Grundlagen).

Sie können diese Links manuell entfernen oder Sie benutzen `update-rc.d` (siehe auch `update-rc.d(8)`). So können Sie zum Beispiel einen Dienst in den Multi-User-Runleveln abschalten:

```
# update-rc.d stop XX 2 3 4 5 .
```

Wobei *XX* eine Zahl ist, die bestimmt, wann die Stop-Aktion für diesen Dienst ausgeführt wird. Bitte beachten Sie, dass `update-rc.d -f Dienst remove` nicht korrekt arbeiten wird, wenn Sie *nicht* `file-rc` benutzen, da *alle* Verknüpfungen entfernt werden. Nach einer Neuinstallation oder einem Upgrade dieses Paketes werden diese Verknüpfungen neu angelegt (was Sie vermutlich nicht wollen). Wenn Sie denken, dass dies nicht sehr intuitiv ist, haben Sie wahrscheinlich recht (siehe Bug 67095 (`http://bugs.debian.org/67095`)). Aus der Manpage:

```
If any files /etc/rcrunlevel.d/[SK]??name already exist then
update-rc.d does nothing. This is so that the system administrator
can rearrange the links, provided that they leave at least one
link remaining, without having their configuration overwritten.
```

[4] Dies ist z.B. wünschenswert, wenn Sie eine Chroot-Umgebung zur Entwicklung einrichten.

Wenn Sie `file-rc` benutzen, werden alle Informationen über das Starten von Diensten durch eine gemeinsame Konfigurationsdatei verarbeitet und sogar nach der Deinstallation von Paketen beibehalten.

Sie können das TUI (Text User Interface, textbasierte Benutzungsoberfläche) des Paketes `sysv-rc-conf` benutzen, um all diese Änderungen einfach zu erledigen (`sysv-rc-conf` arbeitet sowohl mit `file-rc` als auch mit normalen System-V-Runleveln). Es gibt auch vergleichbare GUIs für Desktop-Systeme. Sie können auch die Befehlszeile von `sysv-rc-conf` verwenden:

```
# sysv-rc-conf foobar off
```

Der Vorteil dieses Werkzeugs ist, dass die rc.d-Links wieder auf den Status zurückgesetzt werden, die sie vor dem Aufruf von »off« hatten, wenn Sie den Dienst wieder aktivieren mit:

```
# sysv-rc-conf foobar on
```

Andere (nicht empfohlene) Methoden zum Abschalten eines Dienstes sind:

- Benennen Sie die Skriptdatei (`/etc/init.d/Dienst`) um (zum Beispiel in `/etc/init.d/OFF.Dienst`). Da das Verweise, die auf kein Ziel verweisen (dangling symlinks), erzeugt, werden beim Booten Fehlermeldungen erzeugt werden.
- Entfernen Sie das Ausführungsrecht von der Datei `/etc/init.d/Dienst`. Auch das wird beim Booten Fehlermeldungen verursachen.
- Editieren der Datei `/etc/init.d/Dienst`, so dass sich das Skript sofort beendet, sobald es gestartet wird, indem Sie die Zeile `exit 0` am Anfang einfügen oder den `start-stop-daemon`-Abschnitt auskommentieren. Falls Sie dies tun, können Sie das Skript nicht später dazu verwenden, den Dienst von Hand zu starten.

Jedoch handelt es sich bei allen Dateien unter `/etc/init.d` um Konfigurationsdateien und sollten daher bei einer Paketaktualisierung nicht überschrieben werden.

Sie können im Gegensatz zu anderen (UNIX-)Betriebssystemen Dienste unter Debian nicht abschalten, indem Sie die Dateien unter `/etc/default/Dienst` modifizieren.

FIXME: Add more information on handling daemons using file-rc

3.6.2 Abschalten von `inetd` oder seinen Diensten

Sie sollten überprüfen, ob Sie heutzutage den `inetd`-Daemon überhaupt brauchen. Inetd war früher eine Möglichkeit, Unzulänglichkeiten des Kernels auszugleichen. Diese sind aber in modernen Linux-Kerneln nicht mehr vorhanden. Gegen `inetd` gibt es die Möglichkeit von Angriffen, die zur Dienstverweigerung führen (Denial of Service), welche die Last des Rechners unglaublich erhöhen. Viele Leute ziehen es vor, einzelne Daemonen zu benutzen, anstatt

einen Dienst über `inetd` zu starten. Wenn Sie immer noch einen `inetd`-Dienst laufen lassen wollen, wechseln Sie wenigstens zu einem besser zu konfigurierenden Inet-Daemonen wie `xinetd` oder `rlinetd`.

Sie sollten alle nicht benötigten Inetd-Dienste auf Ihrem System abschalten, wie zum Beispiel `echo`, `chargen`, `discard`, `daytime`, `time`, `talk`, `ntalk` und die r-Dienste (`rsh`, `rlogin` und `rcp`), die als SEHR unsicher gelten (benutzen Sie stattdessen `ssh`).

Sie können Dienste abschalten, indem Sie direkt `/etc/inetd.conf` editieren, aber Debian stellt Ihnen einen besseren Weg zur Verfügung: `update-inetd` (was die Dienste auf eine Art herauskommentiert, so dass sie leicht wieder eingeschaltet werden können). Sie können den `Telnet`-Daemon sehr leicht mit dem folgenden Kommando abschalten, so dass die Konfigurationsdateien angepasst und der Daemon neu gestartet wird:

```
/usr/sbin/update-inetd --disable telnet
```

Wenn Sie Dienste starten wollen, aber nur auf bestimmten IP-Adressen Ihres Systems, müssen Sie eventuell auf eine undokumentierte Funktion des `inetd` zurückgreifen (Austausch des Namens des Dienstes durch dienst@ip). Alternativ können Sie einen Daemon wie `xinetd` benutzen.

3.7 Installieren Sie möglichst wenig Software

Debian bietet *sehr viel* Software an. Debian 3.0 (*Woody*) enthält sechs oder sieben (je nach Architektur) CDs mit Software und tausenden Paketen. Debian 3.1 *Sarge* wird mit etwa 13 CD-ROMs ausgeliefert werden. Bei so viel Software, selbst wenn Sie die Installation auf das Basis-System reduzieren [5], könnten Sie auf Abwege geraten und mehr installieren, als Sie wirklich benötigen.

Da Sie bereits wissen, was Sie mit Ihrem System machen wollen (oder etwa nicht?), sollten Sie nur Software installieren, die Sie wirklich für den Betrieb benötigen. Jedes unnötig installierte Programm könnte von einem Nutzer, der Ihr System kompromittieren will, genutzt werden – oder von einem externen Eindringling, der Shell-Zugriff bekommen hat (oder der Code von außerhalb durch einen fehlerhaften Dienst ausführen kann).

Zum Beispiel kann das Vorhandensein von Hilfsprogrammen für Programmierer (ein C-Compiler) oder Interpretern (wie `Perl` s.u., `Python`, `tcl`, ...) einem Angreifer helfen, das System weiter zu kompromittieren:

[5] Unter Debian-Woody ist das Basis-System etwa 400-500MB groß. Probieren Sie Folgendes:

```
, $ size=0 $ for i in ‘grep -A 1 -B 1 "^Section: base"/var/lib/dpkg/available
| grep -A 2 "^Priority: requiredgrep "^Installed-Sizecut -d : -f 2 ‘; do
size=$(($size+$i)); done $ echo $size 47762
```

- Der Angreifer kann seine Privilegien auf dem System erweitern. Es ist beispielsweise leichter, eine lokale Sicherheitslücke des Systems auszunutzen, wenn man einen Debugger und Compiler zur Verfügung hat, um den eigenen Exploit (ein Programm, das eine Sicherheitslücke ausnutzt) zu kompilieren und zu testen.
- Man könnte dem Angreifer Werkzeuge zur Verfügung stellen, die ihm helfen könnten, das kompromittierte System als *Basis für Angriffe* auf andere Systeme zu benutzen. [6]

Natürlich kann ein Eindringling mit lokalem Shell-Zugriff seine eigenen Programme herunterladen und ausführen. Und sogar die Shell selbst kann benutzt werden, um komplexere Programme zu schreiben. Das Entfernen unnötiger Programme wird also nicht helfen, das Problem zu *verhindern*. Jedoch wird es für den Angreifer wesentlich schwieriger werden, das System zu kompromittieren (und manchmal wird er in dieser Situation aufgeben und sich ein leichteres Ziel suchen). Wenn Sie also auf einem produktivem System Werkzeuge lassen, die benutzt werden können, um andere Systeme anzugreifen (siehe 'Programme zur Fernprüfung der Verwundbarkeit' auf Seite 163), müssen Sie auch davon ausgehen, dass ein Angreifer sie auch benutzen wird.

Beachten Sie bitte, dass eine Standardinstallation von Debian *Sarge* (d.h. eine Installation, bei der nicht individuell Pakete ausgewählt werden) einige Pakete zur Softwareentwicklung installieren wird, die normalerweise nicht benötigt werden. Das liegt daran, dass einige Pakete zur Softwareentwicklung die Priorität *Standard* haben. Wenn Sie keine Software entwickeln, können Sie ohne Bedenken die folgenden Pakete von Ihrem System entfernen, was nebenbei auch etwas Platz schafft:

```
Paket                     Größe
------------------------+--------
gdb                     2.766.822
gcc-3.3                 1.570.284
dpkg-dev                  166.800
libc6-dev               2.531.564
cpp-3.3                 1.391.346
manpages-dev            1.081.408
flex                      257.678
g++                         1.384 (Hinweis: virtuelles Paket)
linux-kernel-headers    1.377.022
bin86                      82.090
cpp                        29.446
gcc                         4.896 (Hinweis: virtuelles Paket)
g++-3.3                 1.778.880
bison                     702.830
make                      366.138
libstdc++5-3.3-dev        774.982
```

[6] Häufig werden fremde Systeme nur deshalb gehackt, weil Sie zu weiteren illegitimen Aktivitäten benutzt werden sollen (DoS-Attacken, Spam, geheime FTP-Server, DNS-Schweinereien, ...). Der Angreifer möchte meist gar nicht an die vertraulichen Daten auf dem kompromittierten System herankommen.

Das ist etwas, was wahrscheinlich in Veröffentlichungen nach Sarge verbessert werden wird. Um den aktuellen Status dieser Sache zu verfolgen, sehen Sie sich Fehler #301273 (http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=301273) und Fehler #301138 (http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=301138) an. Wegen eines Fehlers im Installationssystem ist dies nicht geschehen, wenn mit dem Installationssystem von Debian 3.0 *Woody* installiert wird.

3.7.1 Entfernen von Perl

Sie müssen bedenken, dass es nicht gerade einfach ist, `Perl` von einem Debian-System zu entfernen (in der Tat kann es ziemlich schwierig werden), da es von vielen Dienstprogrammen benutzt wird. `perl-base` hat außerdem *Priority: required* (und das sagt eigentlich schon alles). Es ist aber trotzdem machbar. Allerdings können Sie auf diesem System keine `Perl`-Anwendung mehr laufen lassen. Außerdem müssen Sie auch das Paket-Management-System hereinlegen, damit es weiterhin denkt, dass `perl-base` installiert ist, auch wenn es das nicht mehr ist. [7]

Welche Dienstprogramme benutzen `Perl`? Sie können es selbst herausfinden:

```
$ for i in /bin/* /sbin/* /usr/bin/* /usr/sbin/*; do [ -f $i ] && {
type=‘file $i | grep -il perl‘; [ -n "$type" ] && echo $i; }; done
```

Diese Liste schließt die folgenden Dienstprogramme mit der Priorität *required* oder *important* ein:

- `/usr/bin/chkdupexe` aus dem Paket `util-linux`.
- `/usr/bin/replay` aus dem Paket `bsdutils`.
- `/usr/sbin/cleanup-info` aus dem Paket `dpkg`.
- `/usr/sbin/dpkg-divert` aus dem Paket `dpkg`.
- `/usr/sbin/dpkg-statoverride` aus dem Paket `dpkg`.
- `/usr/sbin/install-info` aus dem Paket `dpkg`.
- `/usr/sbin/update-alternatives` aus dem Paket `dpkg`.
- `/usr/sbin/update-rc.d` aus dem Paket `sysvinit`.
- `/usr/bin/grog` aus dem Paket `groff-base`.
- `/usr/sbin/adduser` aus dem Paket `adduser`.
- `/usr/sbin/debconf-show` aus dem Paket `debconf`.

[7] Sie können (auf einem anderen System) eine Paket-Attrappe mit `equivs` erstellen.

- `/usr/sbin/deluser` aus dem Paket `adduser`.
- `/usr/sbin/dpkg-preconfigure` aus dem Paket `debconf`.
- `/usr/sbin/dpkg-reconfigure` aus dem Paket `debconf`.
- `/usr/sbin/exigrep` aus dem Paket `exim`.
- `/usr/sbin/eximconfig` aus dem Paket `exim`.
- `/usr/sbin/eximstats` aus dem Paket `exim`.
- `/usr/sbin/exim-upgrade-to-r3` aus dem Paket `exim`.
- `/usr/sbin/exiqsumm` aus dem Paket `exim`.
- `/usr/sbin/keytab-lilo` aus dem Paket `lilo`.
- `/usr/sbin/liloconfig` aus dem Paket `lilo`.
- `/usr/sbin/lilo_find_mbr` aus dem Paket `lilo`.
- `/usr/sbin/syslogd-listfiles` aus dem Paket `sysklogd`.
- `/usr/sbin/syslog-facility` aus dem Paket `sysklogd`.
- `/usr/sbin/update-inetd` aus dem Paket `netbase`.

Ohne Perl und solange Sie diese Dienstprogramme nicht in einem Shell-Skript neu schreiben, werden Sie also wahrscheinlich keine Pakete mehr verwalten können (und so das System nicht upgraden können, und das ist *keine gute Idee*).

Wenn Sie sich dazu entschlossen haben, Perl aus dem Debian-Basis-System zu entfernen und ein wenig Freizeit haben, schicken Sie uns doch Fehlerberichte zu den aufgezählten Paketen, die (als ein Patch) einen Ersatz dieser Dienstprogramme als Shell-Skript enthalten.

Wenn Sie wissen wollen, welche Debian-Pakete von Perl abhängen, können Sie Folgendes verwenden:

```
$ grep-available -s Package,Priority -F Depends perl
```

oder

```
$ apt-cache rdepends perl
```

3.8 Lesen Sie Debians Sicherheits-Mailinglisten

Es ist niemals falsch, einen Blick in die debian-security-announce Mailingliste zu werfen, wo Anleitungen und Problemlösungen durch das Debian-Sicherheits-Team bekannt gemacht werden, oder sich an mailto:debian-security@lists.debian.org zu beteiligen, wo Sie an Diskussionen zu allen sicherheitsrelevanten Fragen teilnehmen können.

Um wichtige Warnungen zu Sicherheitsaktualisierungen zu erhalten, senden Sie eine E-Mail an debian-security-announce-request@lists.debian.org (mailto:debian-security-announce-request@lists.debian.org) mit dem Wort "subscribe" in der Betreffzeile. Sie können diese moderierte E-Mail-Liste unter http://www.de.debian.org/MailingLists/subscribe auch über das Web abonnieren.

Diese Mailingliste hat ein sehr geringes Aufkommen, und indem Sie sie abonnieren, werden Sie sofort über Sicherheitsaktualisierungen der Debian-Distribution informiert. Dies erlaubt Ihnen sehr schnell, neue Pakete mit Sicherheitsaktualisierungen herunterzuladen, was sehr wichtig ist, um ein sicheres System zu verwalten (siehe 'Ausführen von Sicherheitsupdates' auf Seite 48 für weitere Details, wie Sie dies machen).

Kapitel 4

Nach der Installation

Wenn das System installiert ist, können Sie es noch weiter absichern, indem Sie einige der in diesem Kapitel beschriebenen Schritte ausführen. Natürlich hängt dies vor allem von Ihrem Setup ab, aber um physischen Zugriff zu verhindern, sollten Sie 'Änderungen im BIOS (noch einmal)' auf Seite 51, 'Ein Passwort für LILO oder GRUB einstellen' auf Seite 51, 'Entfernen des Root-Promptes aus dem Kernel' auf Seite 53, 'Booten von Diskette verbieten' auf Seite 54, 'Einschränkender Konsolen-Zugang' auf Seite 54 und 'System-Neustart von der Konsole aus einschränken' auf Seite 55 lesen.

Bevor Sie sich mit einem Netzwerk verbinden, insbesondere wenn es sich um ein öffentliches Netzwerk handelt, sollten Sie wenigstens eine Sicherheitsaktualisierung (siehe 'Ausführen von Sicherheitsupdates' auf der nächsten Seite) durchführen. Optional können Sie auch einen Schnappschuss Ihres Systems machen (siehe 'Einen Schnappschuss des Systems erstellen' auf Seite 94).

4.1 Abonnement der Security-Announce-Mailingliste von Debian

Um Informationen zu verfügbaren Sicherheitsaktualisierungen und die Debian-Sicherheits-Ankündigungen (DSA) zu erhalten, sollten Sie Debians Security-Announce-Mailingliste abonnieren. Lesen Sie 'Das Sicherheitsteam von Debian' auf Seite 137 für weitere Informationen, wie das Sicherheitsteam von Debian arbeitet. Hinweise, wie man die Mailinglisten von Debian abonniert, finden Sie unter `http://lists.debian.org`.

DSAs werden mit der Signatur des Sicherheitsteams von Debian unterschrieben, die unter `http://security.debian.org` erhältlich ist.

Sie sollten in Betracht ziehen, auch die debian-security Mailingliste (`http://lists.debian.org/debian-security`) zu abonnieren. Dort finden allgemeine Diskussionen zu Sicherheitsthemen im Betriebssystem Debian statt. Sie können auf der Liste sowohl mit gleichgesinnten Systemadministratoren als auch mit Entwicklern von Debian und Programmautoren in Kontakt treten. Diese werden Ihre Fragen beantworten und Ihnen Ratschläge geben.

FIXME: Add the key here too?

4.2 Ausführen von Sicherheitsupdates

Sobald neue Sicherheitslöcher in einem Paket entdeckt wurden, reparieren sie Debians Paketbetreuer und Originalautoren im Allgemeinen innerhalb von Tagen oder sogar Stunden. Nachdem das Loch gestopft wurde, werden neue Pakete unter `http://security.debian.org` bereit gestellt.

Wenn Sie ein Debian-Release installieren, müssen Sie berücksichtigen, dass es seit dem Release Sicherheitsupdates gegeben haben könnte, nachdem entdeckt wurde, dass ein bestimmtes Paket verwundbar ist. Ebenso könnte es kleinere Releases gegeben haben (es gab sieben kleinere Releases von Debian 2.2 *Potato*), die diese Paketaktualisierungen enthalten.

Sie müssen sich das Erstellungsdatum Ihres CD-Sets (wenn Sie ein solches benutzen) notieren und auf der Sicherheitsseite nachprüfen, ob es Sicherheitsaktualisierungen gegeben hat. Wenn es solche gibt, und Sie die Pakete nicht von der Sicherheitsseite mit einem anderen System herunterladen können (Ihr System ist doch nicht schon mit dem Internet verbunden, oder?), könnten Sie es in Erwähnung ziehen (falls Sie nicht beispielsweise durch eine Firewall, geschützt sind), Firewall-Regeln zu aktivieren, so dass Ihr System ausschließlich mit security.debian.org Verbindung aufnehmen kann, und dann ein Update ausführen. Eine Beispielkonfiguration finden Sie unter ‘Schutz der Sicherheitsaktualisierung durch eine Firewall’ auf Seite 247.

*Anmerkung:*Seit Debian Woody 3.0 wird Ihnen nach der Installation die Möglichkeit eingeräumt, Sicherheitsaktualisierungen Ihrem System hinzuzufügen. Wenn Sie hier ’ja’ sagen, wird das Installationssystem die passenden Schritte unternehmen, um die Quellen der Sicherheitsaktualisierungen Ihren Paketquellen hinzuzufügen. Falls Sie nun eine Internetverbindung haben, wird Ihr System alle Sicherheitsaktualisierungen herunterladen, die seit Entstehung Ihres Installationsmediums erzeugt wurden. Falls Sie ein Upgrade von einer älteren Version von Debian durchführen, oder Sie das Installationssystem anweisen, dies nicht zu tun, sollten Sie die hier vorgestellten Schritte unternehmen.

Um Ihr System manuell zu aktualisieren, fügen Sie die folgende Zeile in Ihre `/etc/apt/sources.list` ein. So werden Sie Sicherheitsaktualisierungen automatisch erhalten, wann immer Sie Ihr System aktualisieren.

```
deb http://security.debian.org/debian-security stable/updates main contrib
non-free
```

Hinweis: Falls Sie den *Testing*-Zweig einsetzen, sollten Sie die Sicherheitsspiegel für Testing verwenden. Das wird unter ‘Sicherheitsunterstützung für den Testing-Zweig’ auf Seite 184 beschrieben.

Wenn Sie dies erledigt haben, können Sie Ihr System mit `apt` oder `dselect` aktualisieren.

- Falls Sie `apt` verwenden wollen, müssen Sie (als root) nur Folgendes tun:

```
# apt-get update
# apt-get upgrade
```

- Falls Sie `dselect` verwenden wollen, müssen Sie zuerst aktualisieren ([U] für Update), dann installieren ([I] für Install) und schließlich die installieren/aktualisierten Pakete konfigurieren ([C] für Configure).

Wenn Sie möchten, können Sie ebenfalls eine deb-src Zeile hinzufügen. Weitere Details finden Sie unter `apt(8)`.

Anmerkung: Sie brauchen folgende Zeile *nicht* hinzufügen:

```
deb http://security.debian.org/debian-non-US stable/non-US main contrib
non-free
```

Das liegt daran, dass sich der Server, der security.debian.org hostet, außerhalb der USA befindet und somit kein getrenntes Archiv für Non-US hat.

4.2.1 Sicherheitsaktualisierungen von Bibliotheken

Wenn Sie eine Sicherheitsaktualisierung durchgeführt haben, müssen Sie gegebenenfalls einige Dienste des Systems neu starten. Wenn Sie das nicht tun, könnten Dienste auch nach der Sicherheitsaktualisierung immer noch verwundbar sein. Das liegt daran, dass Daemonen, die schon vor einem Upgrade liefen, immer noch die alten Bibliotheken vor dem Upgrade verwenden könnten.[1] Um herauszufinden, welche Daemonen neu gestartet werden müssen, können Sie das Programm `checkrestart` (ist im Paket `debian-goodies` enthalten) oder diesen Einzeiler (als Root) verwenden:[2]

```
# lsof | grep <aktualisierte_Bibliothek> | awk '{print $1, $9}' | uniq | sort
```

Einige Pakete (wie `libc6`) werden diesen Test in der Postinst-Phase für eine begrenzte Anzahl von Diensten durchführen, da ein Upgrade von notwendigen Bibliotheken einige Anwendungen unbrauchbar machen kann, wenn sie nicht neu gestartet werden [3].

Indem das System auf Runlevel 1 (Single User) und dann zurück auf Runlevel 3 (Multi User) gebracht wird, sollten die meisten (wenn nicht alle) Systemdienste neu gestartet werden. Dies ist aber keine Möglichkeit, wenn Sie die Sicherheitsaktualisierung über eine entfernte Verbindung (z.B. mit ssh) vornehmen, da sie getrennt werden würde.

Lassen Sie Vorsicht walten, wenn Sie es mit Sicherheitsaktualisierungen über eine entfernte Verbindung wie mit ssh zu tun haben. Die empfohlene Vorgehensweise für Sicherheitsaktualisierungen, die Dienste betreffen, ist, den SSH-Daemon neu zu starten und sofort zu versuchen, eine neue SSH-Verbindung herzustellen, ohne die alten zu beenden. Falls der Verbindungsversuch scheitern sollte, machen Sie die Aktualisierung rückgängig und untersuchen Sie das Problem.

[1] Selbst wenn die Bibliotheken aus dem Dateisystem entfernt wurden, werden die Inodes nicht beseitigt, bis kein Programm mehr einen offenen Dateideskriptor mit Verweis auf sie hat.

[2] Je nach der Version von lsof müssen Sie $8 statt $9 verwenden.

[3] Das passierte z.B. beim Upgrade von libc6 2.2.x zu 2.3.x wegen Problemen mit der NSS-Authentifizierung, siehe http://lists.debian.org/debian-glibc/2003/debian-glibc-200303/msg00276.html.

4.2.2 Sicherheitsaktualisierung des Kernels

Stellen Sie zunächst sicher, dass Ihr Kernel durch das Paketsystem verwaltet wird. Wenn Sie die Installation mit dem Installationssystem von Debian 3.0 oder früher durchgeführt haben, ist Ihr Kernel *nicht* in das Paketsystem integriert und könnte veraltet sein. Sie können das leicht überprüfen, indem Sie Folgendes ausführen:

```
$ dpkg -S ‘readlink -f /vmlinuz‘
kernel-image-2.4.27-2-686: /boot/vmlinuz-2.4.27-2-686
```

Wenn Ihr Kernel nicht vom Paketsystem verwaltet wird, werden Sie anstatt der obigen Nachricht die Rückmeldung bekommen, dass das Paketverwaltungsprogramm kein Paket finden konnte, das mit der Datei verbunden ist. Die obige Meldung besagt, dass die Datei, die mit dem laufenden Kernel verbunden ist, vom Paket `kernel-image-2.4.27-2-686` zur Verfügung gestellt wird. Sie müssen also zuerst ein Paket mit einem Kernel-Image von Hand installieren. Das genaue Kernel-Image, das Sie installieren sollten, hängt von Ihrer Architektur und Ihrer bevorzugten Kernelversion ab. Wenn Sie das einmal erledigt haben, können Sie die Sicherheitsaktualisierungen des Kernels wie die jedes anderen Pakets durchführen. Beachten Sie allerdings, dass Kernelaktualisierungen *nur* für Aktualisierungen der gleichen Kernelversion wie der Ihrigen durchgeführt werden. D.h. `apt` wird nicht automatisch Ihren Kernel von 2.4 auf 2.6 aktualisieren (oder von 2.4.26 auf 2.4.27 [4]).

Das Installationssystem von Debian 3.1 wird den gewählten Kernel (2.4 oder 2.6) als Teil des Paketsystems behandeln. Sie können überprüfen, welche Kernel Sie installiert haben:

```
$ COLUMNS=150 dpkg -l ’kernel-image*’ | awk ’$1 ~ /ii/ { print $0 }’
```

Um festzustellen, ob Ihr Kernel aktualisiert werden muss, führen Sie Folgendes aus:

```
$ kernfile=‘readlink -f /vmlinuz‘
$ kernel=‘dpkg -S $kernfile | awk -F : ’{print $1}’‘
$ apt-cache policy $kernel
kernel-image-2.4.27-2-686:
  Installed: 2.4.27-9
  Candidate: 2.4.27-9
  Version Table:
 *** 2.4.27-9 0
(...)
```

Wenn Sie eine Sicherheitsaktualisierung durchführen, die auch das Kernel-Image umfasst, *müssen* Sie das System neu starten, damit die Sicherheitsaktualisierung Wirkung zeigen kann. Anderenfalls lassen Sie immer noch das alte (und verwundbare) Kernel-Image laufen.

[4] Es sei denn, Sie haben ein Kernel-Metapaket wie `kernel-image-2.4-686` installiert, welches immer die neueste Minor-Version des Kernels einer Architektur installieren wird.

Wenn Sie das System neu starten müssen (wegen eines Kernel-Upgrades), sollten Sie sicherstellen, dass der Kernel fehlerfrei booten wird und die Netzwerkverbindungen hergestellt werden, besonders wenn die Sicherheitsaktualisierung über eine entfernte Verbindung wie mit ssh durchgeführt wird. Für den ersten Fall können Sie Ihren Boot-Loader so konfigurieren, dass er den Originalkernel lädt, wenn ein Fehler auftritt (für weiterführende Informationen sollten Sie Remotely rebooting Debian GNU/Linux machines (http://www.debian-administration.org/?article=70) lesen). Im zweiten Fall müssen Sie ein Skript verwenden, das die Netzwerkverbindungen testen kann und überprüft, ob der Kernel das Netzwerksystem korrekt gestartet hat, und, wenn das nicht geschehen ist, das System neu startet [5]. Dies sollte böse Überraschungen verhindern, wie wenn man den Kernel aktualisiert und dann nach einem Reboot merkt, dass die Netzwerkhardware nicht richtig erkannt oder konfiguriert wurde, und man daher eine weite Strecke zurücklegen muss, um das System wieder hoch zu bringen. Natürlich hilft es beim Debuggen von Reboot-Problemen aus der Ferne, wenn die serielle Konsole des Systems [6] mit einem Konsolen- oder Terminalserver verbunden ist.

4.3 Änderungen im BIOS (noch einmal)

Erinnern Sie sich an 'Setzen Sie ein Passwort im BIOS' auf Seite 35? Nun, jetzt sollten Sie, nachdem Sie nicht mehr von austauschbaren Datenträgern booten müssen, die Standard-BIOS-Einstellung so umändern, dass es *ausschließlich* von der Festplatte bootet. Gehen Sie sicher, dass Sie Ihr BIOS-Passwort nicht verlieren, oder Sie werden nicht mehr ins BIOS zurückkehren können, um die Einstellung wieder zu ändern, so dass Sie im Falle eines Festplattenfehlers Ihr System wiederherstellen können, indem Sie zum Beispiel eine CD-ROM benutzen.

Eine andere, weniger sichere, aber bequemere Möglichkeit ist es, das BIOS so einzustellen, dass es von der Festplatte bootet, und nur falls dies fehlschlägt versucht, von austauschbaren Datenträgern zu booten. Übrigens wird dies oft so gemacht, weil viele Leute ihr BIOS-Passwort nur selten benutzen, so dass sie es zu leicht vergessen.

4.4 Ein Passwort für LILO oder GRUB einstellen

Jeder kann sehr einfach eine root-Shell auf Ihrem System bekommen, indem er einfach `<Name-Ihres-Bootimages> init=/bin/sh` am Bootprompt eingibt. Nachdem die Passwörter geändert und das System neu gestartet wurde, hat die Person uneingeschränkten Root-Zugang und kann nach Belieben alles auf Ihrem System machen. Nach dieser Prozedur haben Sie keinen Root-Zugang mehr zu Ihrem System, weil Sie das Root-Passwort nicht kennen.

[5] Ein Beispielskript mit dem Namen testnet (http://www.debian-administration.org/articles/70/testnet) ist im Artikel Remotely rebooting Debian GNU/Linux machines (http://www.debian-administration.org/?article=70) enthalten. Ein ausgereifteres Testskript befindet sich im Artikel Testing network connectivity (http://www.debian-administration.org/?article=128)

[6] Das Einrichten einer seriellen Konsole würde den Rahmen dieses Dokuments sprengen. Informationen dazu finden Sie im Serial HOWTO (http://www.tldp.org/HOWTO/Serial-HOWTO.html) und im Remote Serial Console HOWTO (http://www.tldp.org/HOWTO/Remote-Serial-Console-HOWTO/index.html).

Um sicher zu stellen, dass dies nicht passieren kann, sollten Sie den Boot-Loader mit einem Passwort schützen. Sie können zwischen einem globalen Passwort und Passwörtern für bestimmte Images wählen.

Für LILO müssen Sie die Konfigurationsdatei `/etc/lilo.conf` editieren und eine `password` und `restricted` Zeile, wie im folgenden Beispiel, einfügen:

```
image=/boot/2.2.14-vmlinuz
   label=Linux
   read-only
   password=hackmich
   restricted
```

Gehen Sie danach sicher, dass die Konfigurationsdatei nicht für alle lesbar ist, um zu verhindern, dass lokale Nutzer das Passwort lesen können. Haben Sie dies getan, rufen Sie lilo auf. Wenn Sie die `restricted`-Zeile weglassen, wird lilo immer nach dem Passwort fragen, egal ob LILO Parameter übergeben wurden oder nicht. Die Standard-Zugriffsrechte auf `/etc /lilo.conf` erlauben root das Lesen und Schreiben, und der Gruppe von lilo.conf, ebenfalls root, das Lesen.

Wenn Sie GRUB anstelle von LILO verwenden, editieren Sie `/boot/grub/menu.lst` und fügen die folgenden zwei Zeilen am Anfang an (dabei ersetzen Sie natürlich `hackmich` mit dem vorgesehenen Passwort). Dies verhindert, dass Benutzer die Booteinträge verändern können. `timeout 3` legt eine Wartedauer von 3 Sekunden fest, bevor `grub` den Standard-Eintrag bootet.

```
timeout 3
password hackmich
```

Um die Integrität Ihres Passwortes zusätzlich abzusichern, sollten Sie Ihr Passwort verschlüsselt ablegen. Das Dienstprogramm `grub-md5-crypt` erzeugt ein gehashtes Passwort, das kompatibel mit GRUBs Verschlüsselungsalgorithmus (MD5) ist. Um `grub` mitzuteilen, dass ein Passwort im MD5-Format verwendet wird, benutzen Sie die folgende Anweisung:

```
timeout 3
password --md5 $1$arPydhOM$bIgEKjMW5kxeEuvE1Rah4/
```

Der Parameter –md5 wurde hinzugefügt, um bei `grub` einen MD5-Authentifizierungsprozess zu erzwingen. Das angegebene Passwort ist die MD5 verschlüsselte Version von "hackmich". MD5-Passwörter sind Klartext-Passwörtern vorzuziehen. Weitere Informationen über `grub`-Passwörter können Sie im `grub-doc`-Paket finden.

4.5 Entfernen des Root-Prompts von Initramfs

Hinweis: Dies betrifft alle Standard-Kernel, die nach Debian 3.1 veröffentlicht wurden.

Die Linux-Kernel 2.6 enthalten die Möglichkeit, während des Bootvorgangs eine Root-Shell zu erhalten. Dies geschieht, wenn beim Laden von initramfs ein Fehler auftritt. Dadurch kann der Administrator auf eine Rettungsshell mit Root-Rechten zugreifen. Mit dieser Shell können von Hand Module geladen werden, falls eine automatische Erkennung scheitern sollte. Diese Verhalten ist Standard für ein von `initramfs-tools` erzeugtes Initramfs. Folgende Fehlermeldung wird auftreten:

```
"ALERT!  /dev/sda1 does not exist.  Dropping to a shell!
```

Um dieses Verhalten abzuschalten, müssen Sie folgenden Boot-Parameter setzen: *panic=0*. Sie können ihn entweder in den Abschnitt kopt in `/boot/grub/menu.lst` eintragen und `update-grub` ausführen oder ihn dem Abschnitt append von `/etc/lilo.conf` hinzufügen.

4.6 Entfernen des Root-Promptes aus dem Kernel

Hinweis: Dies trifft nicht auf Kernel zu, die in Debian 3.1 enthalten sind, da die Wartezeit auf Null verändert wurde.

Linux 2.4 Kernel bieten kurz nach dem Laden des cramfs einen Weg Zugriff auf eine Root-Shell zu bekommen, also während das System bootet. Es erscheint eine Meldung, die dem Administrator erlaubt, eine auszuführende Shell mit Root-Privilegien zu öffnen. Diese Shell kann dazu benutzt werden, manuell Module zu laden, falls die automatische Erkennung fehlschlägt. Dies ist das Standard-Verhalten bei `initrd`'s `linuxrc`. Die folgende Meldung wird erscheinen:

```
Press ENTER to obtain a shell (waits 5 seconds)
```

Um dieses Verhalten zu entfernen, müssen Sie `/etc/mkinitrd/mkinitrd.conf` editieren und den Eintrag

```
# DELAY  Anzahl Sekunden, die das linuxrc Skript warten soll,
# um den Nutzer Eingriffe zu erlauben, bevor das System hochgefahren
# wird
DELAY=0
```

setzen.

Erstellen Sie anschließend Ihr Ramdisk-Image neu. Dies können Sie zum Beispiel so tun:

```
# cd /boot
# mkinitrd -o initrd.img-2.4.18-k7 /lib/modules/2.4.18-k7
```

oder (vorzugsweise) so:

```
# dpkg-reconfigure kernel-image-2.4.x-yz
```

Beachten Sie, dass Debian 3.0 *Woody* den Benutzern erlaubt, 2.4er Kernel zu installieren (indem Sie *flavors* wählen), *aber* der Standard-Kernel ist 2.2 (von einigen Architekturen abgesehen, auf die der Kernel 2.2 nicht portiert wurde).

4.7 Booten von Diskette verbieten

Der Standard-MBR von Debian vor Version 2.2 verhielt sich nicht wie ein normaler Master-Boot-Record und ließ eine Methode offen, einfach in ein System einzubrechen:

- Drücken Sie während des Starts Shift, und der MBR-Prompt erscheint.
- Drücken Sie dann F, und Ihr System wird von Diskette booten. Dies kann man ausnutzen, um root-Privilegien auf dem System zu erlangen.

Dieses Verhalten können Sie ändern, indem Sie

```
lilo -b /dev/hda
```

eingeben.

Nun ist LILO in den MBR geschoben worden. Dies können Sie auch erreichen, indem Sie `boot=/dev/hda` zu `lilo.conf` hinzufügen. Es gibt noch eine Möglichkeit, die den MBR komplett abschaltet:

```
install-mbr -i n /dev/hda
```

Auf der anderen Seite könnte diese "Hintertür", die viele Leute nicht kennen, Ihnen einmal die Haut retten, wenn Sie aus irgendwelchen Gründen in große Schwierigkeiten mit Ihrer Installation geraten.

FIXME: Überprüfen, ob das für 2.2 wirklich stimmt, oder war es 2.1? INFO: Die Bootdisketten von Debian 2.2 installieren KEINEN MBR, sondern nur LILO.

4.8 Einschränkender Konsolen-Zugang

Manche Sicherheitsregelwerke könnten Administratoren dazu zwingen, sich erst als Benutzer mit ihrem Passwort auf dem System einzuloggen, und dann Superuser zu werden (mit `su` oder `sudo`). Solche eine Policy ist in Debian in der Datei `/etc/login.defs` oder `/etc/securetty` (falls Sie PAM verwenden) implementiert.

- In `login.defs` ändern Sie die CONSOLE-Variable, die eine Datei oder eine Liste von Terminals definiert, an denen sich Root einloggen darf.
- In `securetty` [7] entfernen Sie oder fügen Sie Terminals hinzu, auf die Root zugreifen darf. Falls Sie nur lokalen Zugang zur Konsole erlauben wollen, benötigen Sie *console*, *ttyX* [8] und *vc/X* (falls Sie die *devfs*-Schnittstelle verwenden). Sie sollten auch *ttySX* [9] hinzufügen, wenn Sie eine serielle Konsole für den lokalen Zugang verwenden. (Wenn X eine ganze Zahl (Integer) ist, sollten Sie mehrere Instanzen [10] haben, abhängig von der Anzahl der virtuellen Konsolen, die Sie in `/etc/inittab` aktiviert haben [11]). Weiterführende Informationen zu Terminal-Schnittstellen finden Sie im Text-Terminal-HOWTO (`http://tldp.org/HOWTO/Text-Terminal-HOWTO-6.html`).

Wenn Sie PAM benutzen, können Sie auch andere Änderungen am Login-Prozess, die auch Einschränkungen für einzelne Benutzer oder Gruppen zu bestimmten Zeiten enthalten können, durch Konfiguration der Datei `/etc/pam.d/login` vornehmen. Eine interessante Eigenschaft, die man auch abschalten kann, ist die Möglichkeit, sich mit einem leeren Passwort (Null Passwort) einzuloggen. Diese Eigenschaft kann eingeschränkt werden, indem sie *nullok* aus der Zeile

```
auth       required   pam_unix.so nullok
```

entfernen.

4.9 System-Neustart von der Konsole aus einschränken

Wenn an Ihr System eine Tastatur angeschlossen ist, kann jeder (ja wirklich *jeder*) Ihr System neu starten, ohne sich in Ihr System einloggen zu müssen. Dies könnte gegen Ihre Sicherheitsrichtlinien verstoßen (oder auch nicht). Wenn Sie dies einschränken wollen, müssen Sie in `/etc/inittab` prüfen, ob die Zeile, die enthält, dass `ctrlaltdel shutdown` aufruft, den `-a` Schalter enthält (vergessen Sie nicht, `init q` auszuführen, nachdem Sie diese Datei irgendwie verändert haben). Standardmäßig enthält Debian diesen Schalter:

```
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
```

Jetzt müssen Sie, um es *manchen* Benutzern zu erlauben, Ihr System neu zu starten, eine Datei `/etc/shutdown.allow` erstellen, wie es die Handbuchseite `shutdown(8)` beschreibt. Dort

[7] Die Datei `/etc/securetty` ist eine Konfigurationsdatei, die zum Paket `login` gehört.
[8] Oder *ttyvX* in GNU/FreeBSD und *ttyE0* in GNU/KNetBSD.
[9] Oder *comX* in GNU/Hurd, *cuaaX* in GNU/FreeBSD und *ttyXX* in GNU/KNetBSD.
[10] Die Standardeinstellung in *Woody* beinhaltet zwölf lokale tty- und vc-Konsolen und die *console*-Schnittstelle. Anmeldungen von entfernten Orten sind nicht erlaubt. In *Sarge* stellt die Standardeinstellung 64 Konsolen für tty- und vc-Konsolen zu Verfügung. Sie können das ohne Probleme entfernen, wenn Sie nicht derartige viele Konsolen benutzen.
[11] Achten Sie auf die *getty* Einträge.

müssen die Namen der Benutzer, die das System neu booten dürfen, aufgeführt sein. Wenn der *drei Finger Salut* (auch bekannt als *Strg+Alt+Entf* und *Affengriff*) ausgeführt wird, wird geprüft, ob irgendeiner der Benutzer, die in der Datei aufgelistet sind, eingeloggt sind. Wenn es keiner von ihnen ist, wird `shutdown` das System *nicht* neu starten.

4.10 Partitionen auf die richtige Art einbinden

Wenn Sie eine ext2-Partition einbinden, können Sie verschiedene Optionen mit dem mount-Befehl oder in `/etc/fstab` verwenden. Dies ist zum Beispiel mein fstab-Eintrag für meine `/tmp`-Partition:

```
/dev/hda7    /tmp    ext2    defaults,nosuid,noexec,nodev    0    2
```

Sie sehen den Unterschied in der Spalte mit den Optionen. Die Option `nosuid` ignoriert komplett alle setuid- und setgid-Bits, während `noexec` das Ausführen von Programmen unterhalb des Einhängepunkts verbietet und `nodev` Gerätedateien ignoriert. Das hört sich toll an, aber:

- dies ist nur auf ext2-Dateisysteme anwendbar
- kann leicht umgangen werden

Die `noexec`-Option, die verhindert, dass Binarys ausgeführt werden können, ließe sich in früheren Kernelversionen leicht umgehen:

```
alex@joker:/tmp# mount | grep tmp
/dev/hda7 on /tmp type ext2 (rw,noexec,nosuid,nodev)
alex@joker:/tmp# ./date
bash: ./date: Permission denied
alex@joker:/tmp# /lib/ld-linux.so.2 ./date
Sun Dec  3 17:49:23 CET 2000
```

Neuere Versionen des Kernels verarbeiten aber die Option `noexec` richtig:

```
angrist:/tmp# mount | grep /tmp
/dev/hda3 on /tmp type ext3 (rw,noexec,nosuid,nodev)
angrist:/tmp# ./date
bash: ./tmp: Keine Berechtigung
angrist:/tmp# /lib/ld-linux.so.2 ./date
./date: error while loading shared libraries: ./date: failed to map segment
from shared object: Operation not permitted
```

Wie auch immer, viele Skript-Kiddies haben Exploits, die versuchen eine Datei in `/tmp` zu erstellen und auszuführen. Wenn sie keine Ahnung haben, werden sie in dieser Grube hängen bleiben. Mit anderen Worten: Ein Benutzer kann nicht hereingelegt werden, einen ausführbaren Trojaner in `/tmp` laufen zu lassen, zum Beispiel indem er zufällig `/tmp` in seinen Suchpfad aufnimmt.

Seien Sie sich auch bewusst, dass manche Skripte darauf aufbauen, dass `/tmp` ausführbare Rechte hat. Bemerkenswerterweise hatte (oder hat?) Debconf Probleme bei dieser Sache, weitere Informationen enthält Fehler 116448 (`http://bugs.debian.org/116448`).

Nachfolgend ein gründlicheres Beispiel. Eine Anmerkung dazu: `/var` könnte auch noexec enthalten, aber manche Software [12] verwahrt ihre Programme unterhalb von `/var`. Dasselbe gilt für die nosuid-Option.

```
/dev/sda6   /usr          ext3    defaults,ro,nodev       0       2
/dev/sda12  /usr/share    ext3    defaults,ro,nodev,nosuid        0       2
/dev/sda7   /var          ext3    defaults,nodev,usrquota,grpquota 0      2
/dev/sda8   /tmp          ext3    defaults,nodev,nosuid,noexec,usrquota,grpqu
/dev/sda9   /var/tmp      ext3    defaults,nodev,nosuid,noexec,usrquota,grpqu
/dev/sda10  /var/log      ext3    defaults,nodev,nosuid,noexec    0       2
/dev/sda11  /var/account  ext3    defaults,nodev,nosuid,noexec    0       2
/dev/sda13  /home         ext3    rw,nosuid,nodev,exec,auto,nouser,async,usrq
/dev/fd0    /mnt/fd0      ext3    defaults,users,nodev,nosuid,noexec      0
/dev/fd0    /mnt/floppy   vfat    defaults,users,nodev,nosuid,noexec      0
/dev/hda    /mnt/cdrom    iso9660 ro,users,nodev,nosuid,noexec            0
```

4.10.1 `/tmp` noexec setzen

Seien Sie vorsichtig, wenn Sie `/tmp` noexec setzen und neue Software installieren wollen, da manche Software es während der Installation benutzt. `apt` ist ein solches Programm (siehe `http://bugs.debian.org/116448`), wenn nicht `APT::ExtractTemplates::TempDir` (siehe `apt-extracttemplates(1)`) passend konfiguriert wurde. Sie können diese Variable in `/etc/apt/apt.conf` auf ein anderes Verzeichnis als `/tmp` mit exec-Privilegien setzen.

4.10.2 Setzen von /usr auf nur-lesen

Wenn Sie auf `/usr` nur lesenden Zugriff erlauben, werden Sie nicht in der Lage sein, neue Pakete auf Ihrem Debian GNU/Linux System zu installieren. Sie werden es erst mit Schreibzugriff erneut mounten müssen, die Pakete installieren und dann wieder nur mit lesendem Zugriff mounten. `apt` kann so konfiguriert werden, dass Befehle vor und nach dem Installieren von Paketen ausgeführt werden. Vielleicht sollten Sie es passend konfigurieren.

Dazu öffnen Sie `/etc/apt/apt.conf` und fügen Sie Folgendes ein:

[12] Einiges davon trifft auf den Paketverwalter `dpkg` zu, da die Installations- oder Deinstallationsanweisungen (post, pre) unter `/var/lib/dpkg/` liegen, und auch auf Smartlist.

```
DPkg
{
    Pre-Invoke  { "mount /usr -o remount,rw" };
    Post-Invoke { "mount /usr -o remount,ro" };
};
```

Beachten Sie, dass das Post-Invoke mit einer "/usr busy" Fehlermeldung scheitern wird. Dies passiert vorwiegend, wenn Sie eine Datei benutzen, die aktualisiert wurde. Sie können diese Programme finden, indem Sie

```
# lsof +L1
```

ausführen.

Halten Sie diese Programme an oder starten Sie sie erneut und rufen dann Post-Invoke manuell auf. *Achtung!* Das bedeutet, dass Sie wahrscheinlich jedes Mal Ihre Sitzung von X (falls Sie eine laufen haben) neu starten müssen, wenn Sie ein größeres Upgrade Ihres Systems durchführen. Sie müssen entscheiden, ob ein nur lesbares `/usr` zu Ihrem System passt. Vergleichen Sie auch diese Diskussion auf debian-devel über nur-lesbares /usr (http://lists.debian.org/debian-devel/2001/11/threads.html#00212).

4.11 Den Benutzern einen sicheren Zugang bereitstellen

4.11.1 Nutzerauthentifizierung: PAM

PAM (Pluggable Authentication Modules) erlauben dem Systemadministrator auszuwählen, wie eine Anwendung Benutzer authentifiziert. Beachten Sie, dass PAM nichts machen kann, solange die Anwendung nicht mit Unterstützung für PAM kompiliert wurde. Die meisten Anwendungen, die mit Debian geliefert werden, haben diese Unterstützung eingebaut. Vor Version 2.2 hatte Debian keine Unterstützung für PAM. Die derzeitige Standardkonfiguration für jeden Dienst, der PAM benutzt, ist es, UNIX-Authentifizierung zu emulieren (lesen Sie `/usr/share/doc/libpam0g/Debian-PAM-MiniPolicy.gz`, um mehr darüber zu erfahren, wie PAM-Dienste unter Debian arbeiten *sollten*).

Jede Anwendung mit PAM-Unterstützung stellt eine Konfigurationsdatei unter `/etc/pam.d/` zur Verfügung, in welcher Sie ihr Verhalten einstellen können:

- welches Verfahren zur Authentifizierung benutzt wird.
- welches Verfahren innerhalb einer Sitzung benutzt wird.
- wie Passwörter überprüft werden.

Die folgende Beschreibung ist weit davon entfernt, komplett zu sein. Für weitere Informationen können Sie The Linux-PAM System Administrator's Guide (http://www.kernel.org/

pub/linux/libs/pam/Linux-PAM-html/pam.html) (auf der PAM-Hauptseite (http://www.kernel.org/pub/linux/libs/pam/)) lesen, diese Dokumentation ist auch in dem Paket libpam-doc enthalten.

PAM bieten Ihnen die Möglichkeit, durch mehrere Authentifizierungsschritte auf einmal zu gehen, ohne dass der Benutzer es weiß. Sie können gegen eine Berkeley Datenbank und gegen die normale passwd Datei authentifizieren, und der Benutzer kann sich nur einloggen, wenn er beide Male korrekt authentifiziert wurde. Sie können viel einschränken mit PAM, genauso wie Sie Ihr System weit öffnen können. Seien Sie also vorsichtig. Eine typische Konfigurationszeile hat ein Kontrollfeld als zweites Element. Generell sollte es auf requisite gesetzt werden, so wird ein Loginfehler erzeugt, wenn eines der Module versagt.

Die erste Sache, die ich gerne mache, ist, MD5-Unterstützung zu den PAM-Anwendungen hinzuzufügen, da dies gegen lexikalische Attacken hilft (da Passwörter länger sein können, wenn sie MD5 benutzen). Die folgenden zwei Zeilen sollten Sie in allen Dateien unterhalb von /etc/pam.d/ hinzufügen, die Zugriff auf Ihre Maschine gewähren, wie zum Beispiel login und ssh.

```
# Gehen Sie sicher, dass Sie libpam-cracklib zuerst installiert haben,
# sonst werden Sie sich nicht einloggen können
password   required     pam_cracklib.so retry=3 minlen=12 difok=3
password   required     pam_unix.so use_authtok nullok md5
```

Also, was macht diese Beschwörungsformel nun genau? Die erste Zeile lädt das cracklib PAM-Modul, welches einen Passwort-Sicherheitscheck bereitstellt. Es fragt nach einem neuen Passwort mit mindestens 12 Zeichen, einer Differenz von mindestens 3 Zeichen zum alten Passwort, und erlaubt 3 Versuche. Cracklib benötigt ein Paket mit Wörterlisten (wie wngerman, wenglish, wspanish, ...). Stellen Sie also sicher, dass Sie ein passendes Paket für Ihre Sprache installiert haben. Ansonsten ist cracklib nicht verwendbar. [13] Die zweite Zeile führt das Standardauthentifizierungsmodul mit MD5-Passwörtern aus und erlaubt Passwörter mit einer Länge von Null. Die use_authtok-Anweisung ist notwendig, um das Passwort von dem vorherigen Modul übergeben zu bekommen.

Um sicher zu stellen, dass sich der Benutzer root nur von lokalen Terminals einloggen kann, sollten Sie die folgende Zeile in /etc/pam.d/login eingefügt werden:

```
auth     requisite  pam_securetty.so
```

Danach sollten die Liste der Terminal in /etc/securetty ändern, auf denen sich Root unmittelbar anmelden darf. Alternativ dazu können Sie auch das pam_access-Modul aktivieren und /etc/security/access.conf bearbeiten. Dieses Vorgehen erlaubt eine allgemeinere und feiner abgestimmte Zugangskontrolle, leider fehlen aber vernünftige Protokollmeldungen (diese sind in PAM nicht standardisiert und sind ein besonders unbefriedigendes Problem). Wir werden zu access.conf in Kürze zurückkehren.

[13] Diese Abhängigkeit ist allerdings im Debian 3.0 Paket nicht gelöst. Lesen Sie dazu Bug #112965 (http://bugs.debian.org/112965).

Zu guter Letzt sollte die folgende Zeile in `/etc/pam.d/login` aktiviert werden, um den Benutzern Grenzen ihrer Systemressourcen zu setzen.

```
session  required   pam_limits.so
```

Dies schränkt die Systemressourcen ein, die ein Benutzer nutzen darf (siehe 'Ressourcen-Nutzung limitieren: Die Datei `limits.conf`' auf der nächsten Seite). Sie können zum Beispiel die Anzahl der Logins, die man haben kann, einschränken (für eine Gruppe von Nutzern oder systemweit), die Anzahl der Prozesse, den belegten Speicher etc.

Editieren Sie nun `/etc/pam.d/passwd` und ändern Sie die erste Zeile. Sie sollten die Option "md5" zufügen, um MD5-Passwörter zu benutzen, ändern Sie die minimale Passwort-Länge von 4 auf 6 (oder mehr) und setzen Sie eine Maximallänge, wenn Sie möchten. Die resultierende Zeile wird in etwa so aussehen:

```
password   required   pam_unix.so nullok obscure min=6 max=11 md5
```

Wenn Sie su schützen möchten, so dass nur manche Leute es benutzen können, um root auf Ihrem System zu werden, müssen Sie eine neue Gruppe "wheel" zu Ihrem System hinzufügen (das ist der sauberste Weg, da keine Datei solche Gruppenrechte bisher benutzt). Fügen Sie root und die anderen Benutzer, die zu root `su`en können sollen, zu dieser Gruppe. Fügen Sie anschließend die folgende Zeile zu `/etc/pam.d/su/` hinzu:

```
auth        requisite   pam_wheel.so group=wheel debug
```

Dies stellt sicher, dass nur Personen aus der Gruppe "wheel" `su` benutzen können, um root zu werden. Andere Benutzer wird es nicht möglich sein, root zu werden. Tatsächlich werden Sie eine ablehnende Nachricht bekommen, wenn Sie versuchen root zu werden.

Wenn Sie es nur bestimmten Nutzern erlauben wollen, sich bei einem PAM-Dienst zu authentifizieren, ist dies sehr leicht zu erreichen, indem Sie Dateien benutzen, in denen die Nutzer, denen es erlaubt ist, sich einzuloggen (oder nicht), gespeichert sind. Stellen Sie sich vor, Sie möchten lediglich dem Nutzer 'ref' erlauben, sich mittels `ssh` einzuloggen. Sie schreiben ihn also in eine Datei `/etc/ssh-users-allowed` und schreiben das Folgende in `/etc/pam.d/ssh`:

```
auth        required    pam_listfile.so item=user sense=allow file=/etc/ssh
```

Da es eine Reihe von Sicherheitslücken mit so genannten unsicheren temporären Dateien zum Beispiel in thttpd (vgl. DSA-883-1 (http://www.debian.org/security/2005/dsa-883)) gab, lohnt es sich, das Paket `libpam-tmpdir` zu installieren. Alles, was Sie machen müssen, ist, Folgendes zu `/etc/pam.d/common-session` hinzuzufügen:

```
session    optional     pam_tmpdir.so
```

Es gab auch eine Diskussion, dies standardmäßig in Etch einzufügen. Sehen Sie sich http://lists.debian.org/debian-devel/2005/11/msg00297.html für weitere Informationen an.

Zuletzt, aber nicht am unwichtigsten, erstellen Sie /etc/pam.d/other mit den folgenden Zeilen:

```
auth     required       pam_securetty.so
auth     required       pam_unix_auth.so
auth     required       pam_warn.so
auth     required       pam_deny.so
account  required       pam_unix_acct.so
account  required       pam_warn.so
account  required       pam_deny.so
password required       pam_unix_passwd.so
password required       pam_warn.so
password required       pam_deny.so
session  required       pam_unix_session.so
session  required       pam_warn.so
session  required       pam_deny.so
```

Diese Zeilen stellen für alle Anwendungen, die PAM unterstützen, eine gute Standard-Konfiguration dar (Zugriff wird standardmäßig verweigert).

4.11.2 Ressourcen-Nutzung limitieren: Die Datei limits.conf

Sie sollten sich wirklich ernsthaft mit dieser Datei beschäftigen. Hier können Sie Ihren Benutzern Ressourcen-Limits definieren. In alten Veröffentlichungen war die Konfigurationsdatei /etc/limits.conf. Aber in neueren Versionen (mit PAM) sollte stattdessen die Konfigurationsdatei /etc/security/limits.conf benutzt werden.

Wenn Sie die Ressourcennutzung nicht einschränken, kann *jeder* Nutzer mit einer gültigen Shell auf Ihrem System (oder sogar ein Einbrecher, der das System durch einen Dienst kompromittierte, oder ein außer Kontrolle geratener Daemon) so viel CPU, Speicher, Stack etc. benutzen, wie das System zur Verfügung stellen kann. Dieses Problem der *Überbeanspruchung von Ressourcen* kann mit der Nutzung von PAM gelöst werden.

Es gibt einen Weg, Ressourcen-Limits zu manchen Shells hinzuzufügen (zum Beispiel hat bash ulimit, siehe bash(1)). Aber da nicht alle die gleichen Limits zur Verfügung stellen, und da der Nutzer seine Shell ändern kann (siehe chsh(1)), ist es besser, die Limits in den PAM-Modulen zu platzieren, da diese unabhängig von der verwendeten Shell Anwendung finden und auch PAM-Module betreffen, die nicht shellorientiert sind.

Ressourcen-Limits werden vom Kernel verhängt, aber sie müssen durch limits.conf konfiguriert werden, und die PAM-Konfiguration der verschiedenen Dienste muss das passende PAM laden. Sie können herausfinden, welche Dienste Limits durchsetzen, indem Sie Folgendes ausführen:

```
$ find /etc/pam.d/ \! -name "*.dpkg*" | xargs -- grep limits |grep -v ":#"
```

Für gewöhnlich nehmen `login`, `ssh` und die grafischen Sitzungsmanager (`gdm`, `kdm` und `xdm`) Nutzerlimits in Anspruch, aber Sie sollte dies auch in anderen Konfigurationsdateien für PAM wie für `cron` tun, um zu verhindern, dass Systemdaemonen alle Systemressourcen aufbrauchen.

Die konkreten Begrenzungen, die Sie festlegen wollen, hängt von den Ressourcen Ihres Systems ab. Das ist einer der Hauptgründe, warum keine Limits in der Standardinstallation enthalten sind.

Zum Beispiel nimmt die Konfiguration im Beispiel unten eine Begrenzung von 100 Prozessen für alle Nutzer vor (um *Fork-Bomben* [14] zu vermeiden) und eine Begrenzung auf 10MB Speicher pro Prozess und ein Limit von 10 gleichzeitigen Logins. Benutzer in der Gruppe `adm` haben höhere Begrenzungen und können Dateien mit einem Speicherabbild schreiben, wenn sie das wollen (es gibt also nur eine *weiche* Begrenzung).

```
*               soft    core            0
*               hard    core            0
*               hard    rss             1000
*               hard    memlock         1000
*               hard    nproc           100
*               -       maxlogins       1
*               hard    data            102400
*               hard    fsize           2048
@adm            hard    core            100000
@adm            hard    rss             100000
@adm            soft    nproc           2000
@adm            hard    nproc           3000
@adm            hard    fsize           100000
@adm            -       maxlogins       10
```

Dies könnten die Limits eines Standardnutzers (einschließlich der Systemdaemonen) sein:

```
$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) 102400
file size               (blocks, -f) 2048
max locked memory       (kbytes, -l) 10000
max memory size         (kbytes, -m) 10000
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
```

[14] Programme, die immer mehr Prozesse erzeugen, um so das System zum Absturz zu bringen, d.Ü.

```
max user processes              (-u) 100
virtual memory          (kbytes, -v) unlimited
```

Und dies die Limits für einen Administrator:

```
$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) 102400
file size               (blocks, -f) 100000
max locked memory       (kbytes, -l) 100000
max memory size         (kbytes, -m) 100000
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 2000
virtual memory          (kbytes, -v) unlimited
```

Für mehr Informationen hierzu lesen Sie:

- PAM reference guide for available modules (http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam-6.html)
- PAM configuration article (http://www.samag.com/documents/s=1161/sam0009a/0009a.htm).
- Seifried's Securing Linux Step by Step (http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html) in dem *Limiting users overview* Abschnitt.
- LASG (http://seifried.org/lasg/users/) in dem *Limiting and monitoring users* Abschnitt.

4.11.3 Aktionen bei der Benutzeranmeldung: Editieren von `/etc/login.defs`

Der nächste Schritt ist es, die grundlegende Konfiguration und die Aktionen bei User-Login zu editieren. Beachten Sie, dass diese Datei kein Bestandteil der PAM-Konfiguration ist. Sie ist eine Konfigurationsdatei, die von den Programmen `login`- und `su` berücksichtigt wird. Es ist also wenig sinnvoll, sie auf Fälle abzustimmen, in denen keines der beiden Programme wenigstens indirekt aufgerufen wird (Das Programm `getty`, welches auf der Konsole läuft und die anfängliche Loginaufforderung zu Verfügung stellt, *ruft* `login` auf).

```
FAIL_DELAY          10
```

Diese Variable sollte auf einen höheren Wert gesetzt werden, um es schwerer zu machen, mittels Brute Force (roher Gewalt, stures Durchprobieren, Anm. d. Übers.) auf einem Terminal einzuloggen. Wenn ein falsches Passwort eingegeben wird, muss der potenzielle Angreifer (aber auch der normale Benutzer!) 10 Sekunden warten, um einen neuen login Prompt zu bekommen, was auf die Dauer viel Zeit kostet, wenn sie Passwörter durch testen. Beachten Sie jedoch die Tatsache, dass diese Einstellung nutzlos ist, wenn Sie ein anderes Programm als `getty` benutzen, wie zum Beispiel `mingetty`.

```
FAILLOG_ENAB        yes
```

Wenn Sie diese Variable einschalten, werden fehlgeschlagene Logins protokolliert. Es ist wichtig, hier auf dem Laufendem zu bleiben, um jemanden zu fassen, der eine Brute Force Attacke versucht.

```
LOG_UNKFAIL_ENAB    yes
```

Wenn Sie die Variable *FAILLOG_ENAB* auf yes gesetzt haben, dann sollten Sie auch diese Variable auf yes setzen. Dies wird auch unbekannte Nutzernamen protokollieren. Wenn Sie dies tun, gehen Sie auch sicher, dass die Protokolle sinnvolle Zugriffsrechte haben (zum Beispiel 640, mit einer angemessenen Gruppen-Zugehörigkeit wie adm), weil Nutzer oft versehentlich ihr Passwort als Usernamen eingeben und dies anderen nicht zugänglich machen wollen.

```
SYSLOG_SU_ENAB      yes
```

Dies schaltet das Mitprotokollieren von `su`-Versuchen im `Syslog` ein. Sehr wichtig auf ernsthaften Maschinen, aber beachten Sie, dass dies auch die Privatsphäre verletzen kann.

```
SYSLOG_SG_ENAB      yes
```

Das gleiche wie bei *SYSLOG_SU_ENAB*, jedoch für das Programm `sg`.

```
MD5_CRYPT_ENAB      yes
```

Wie bereits erklärt, reduzieren MD5-Summen-Passwörter großartig das Problem lexikalischer Attacken, da Sie längere Passwörter benutzen können. Wenn Sie slink benutzen, lesen Sie die Dokumentation zu MD5, bevor Sie diese Option einschalten. Ansonsten wird dies in PAM gesetzt.

```
PASS_MAX_LEN        50
```

Wenn Sie MD5-Passwörter in Ihrer PAM Konfiguration aktiviert haben, dann sollten Sie diese Variable auf denselben Wert setzen, den Sie dort benutzt haben.

4.11.4 ftp Einschränken: Editieren von `/etc/ftpusers`

Die Datei `/etc/ftpusers` enthält eine Liste von allen Nutzern, denen es *nicht* erlaubt ist, sich auf dem Rechner mit ftp einzuloggen. Benutzen Sie diese Datei nur, wenn Sie wirklich ftp erlauben wollen (wozu im Allgemeinen nicht geraten wird, da es Klartext-Passwörter benutzt). Wenn Ihr ftp-Daemon PAM unterstützt, können Sie dies ebenfalls benutzen, um Nutzern bestimmte Dienste zu erlauben oder zu verbieten.

FIXME (FEHLER): Ist es ein Fehler, dass `ftpusers` in Debian standardmäßig *nicht* die Benutzer mit Administratorenrecht (in `base-passwd`) beinhaltet?

Folgender Befehl ist ein einfacher Weg, alle Systemkonten zu `/etc/ftpusers` hinzuzufügen:

```
$ awk -F : '{if ($3<1000) print $1}' /etc/passwd > /etc/ftpusers
```

4.11.5 Verwendung von su

Wenn es wirklich benötigt wird, dass Nutzer der Super-User (also Root, d.Ü.) auf Ihrem System werden, zum Beispiel um Pakete zu installieren oder neue Benutzer anzulegen, können Sie das Programm `su` benutzen, um Ihre Identität zu wechseln. Sie sollten jeden Login als Nutzer Root vermeiden und stattdessen das Programm `su` benutzen. Eigentlich ist die beste Lösung, `su` zu entfernen und zu `sudo` zu wechseln, da es eine feinere Steuerung und mehr Möglichkeiten bietet als `su`. Wie auch immer, `su` ist verbreiteter und wird auf vielen Unices benutzt.

4.11.6 Verwendung von sudo

Das `sudo` erlaubt es dem Benutzer, ein definiertes Kommando unter einer anderen Nutzeridentität auszuführen, sogar als Root. Wenn der Nutzer zu `/etc/sudoers` hinzugefügt ist und sich korrekt authentifiziert, ist er in der Lage, Kommandos, die in `/etc/sudoers` definiert wurden, auszuführen. Sicherheitsverletzungen, wie ein inkorrektes Passwort oder der Versuch ein Programm auszuführen, für das Ihre Rechte nicht ausreichen, werden protokolliert und an root gemailt.

4.11.7 Administrativen Fernzugriff verweigern

Sie sollten `/etc/security/access.conf` ebenfalls so verändern, dass ein Login aus der Ferne in ein administratives Konto nicht erlaubt wird. Auf diese Weise müssen die Nutzer das Programm `su` (oder `sudo`) aufrufen, um Administratorenrechte zu bekommen, so dass es immer eine nachprüfbare Spur gibt.

Sie müssen die folgende Zeile zu Ihrer `/etc/security/access.conf` hinzufügen, die Debians Standardkonfigurationsdatei hat ein Beispiel auskommentiert:

```
-:wheel:ALL EXCEPT LOCAL
```

Vergessen Sie nicht, in `/etc/pam.d/` das `pam_access`-Module für jeden Dienst (oder jede Standardkonfiguration) anzuschalten, wenn Sie wollen, dass Ihre Änderungen an `/etc/security/access.conf` berücksichtigt werden.

4.11.8 Den Nutzerzugang einschränken

Manchmal werden Sie vielleicht denken, dass Sie einen Nutzer auf Ihrem System erstellen müssen, um einen bestimmten Dienst (pop3 E-Mail Server oder ftp) anzubieten. Bevor Sie dies tun, erinnern Sie sich zuerst daran, dass die PAM Implementierung in Debian GNU/Linux Ihnen erlaubt, Nutzer mit einer breiten Auswahl von externen Verzeichnisdiensten (radius, ldap, etc.) zu bestätigen. Dies wird vom libpam-Paket bewerkstelligt.

Wenn Sie einen Nutzer anlegen müssen, und auf Ihr System aus der Ferne zugegriffen werden kann, beachten Sie, dass es Nutzern möglich sein wird, sich einzuloggen. Sie können dies beheben, indem Sie diesen Nutzern eine Null (`/dev/null`) als Shell (sie müsste in `/etc/shells` aufgelistet sein) zuweisen. Wenn Sie den Nutzern erlauben wollen, auf das System zuzugreifen, aber ihre Bewegungen einschränken wollen, können Sie `/bin/rbash` benutzen. Dies hat das gleiche Ergebnis, wie wenn Sie die `-r` Option der `Bash` (*RESTRICTED SHELL*, siehe `bash(1)`) verwendet hätten. Beachten Sie bitte, dass sogar mit einer beschränkten Shell ein Nutzer, der auf ein interaktives Programm zugreifen kann (das ihm erlaubt, eine Subshell auszuführen), diese Limitierung der Shell umgehen kann.

Debian bietet zurzeit in seiner Unstable-Veröffentlichung (und wird es vielleicht der nächsten Stable-Veröffentlichung hinzufügen) das `pam_chroot`-Modul (in `libpam-chroot`) an. Eine Alternative hierzu ist es, die Dienste, die eine Fernanmeldung ermöglichen (`ssh` und `telnet`), in einer `chroot`-Umgebung laufen zu lassen. [15]

Wenn Sie einschränken wollen, *wann* ein Nutzer auf das System zugreifen kann, müssen sie `/etc/security/access.conf` an Ihre Bedürfnisse anpassen.

Informationen, wie man Benutzer, die auf das System mittels dem `ssh`-Dienst zugreifen, in eine `chroot`-Umgebung einsperrt, wird in '`Chroot`-Umgebung für `SSH`' auf Seite 249 beschrieben.

4.11.9 Überprüfen der Nutzer

Wenn Sie wirklich paranoid sind, sollten Sie vielleicht eine systemweite Einrichtung verwenden, um zu überwachen, was die Benutzer auf Ihrem System tun. In diesem Abschnitt werden eine Tipps vorgestellt, wie Sie verschiedene Werkzeuge verwenden.

Überwachung von Ein- und Ausgabe mittels eines Skripts

Um sowohl die von den Nutzern ausführten Programme als auch deren Ergebnisse zu überwachen, können Sie den Befehl `script` verwenden. Sie können `script` nicht als eine Shell

[15] `libpam-chroot` wurden noch nicht vollständig getestet. Es funktioniert mit `login`, aber es dürfte nicht leicht sein, diese Umgebung für andere Programme einzurichten.

einsetzen (auch dann nicht, wenn Sie es zu `/etc/shells` hinzufügen). Aber Sie können in die Datei, welche den Startvorgang der Shell steuert, folgendes eintragen:

```
umask 077
exec script -q -a "/var/log/sessions/$USER"
```

Die History-Datei der Shell benutzen

Wenn Sie auswerten wollen, was die Benutzer in die Shell eingeben (aber nicht was das Ergebnis ist), können Sie eine systemweite `/etc/profile` so einrichten, dass alle Befehle in der `History`-Datei (Verlaufsdatei) gespeichert werden. Die systemweite Einstellung muss so eingerichtet werden, dass Benutzer die Auditmöglichkeit nicht aus ihrer Shell entfernen können. Ob dies möglich ist, hängt von der Art der Shell ab. Sie müssen also sicherstellen, dass alle Benutzer eine Shell verwenden, die das unterstützt.

Für die Bash zum Beispiel könnte `/etc/profile` folgendermaßen aufgebaut werden [16] :

```
HISTFILE=~/.bash_history
HISTSIZE=10000
HISTFILESIZE=999999
# Don't let the users enter commands that are ignored
# in the history file
HISTIGNORE=""
HISTCONTROL=""
readonly HISTFILE
readonly HISTSIZE
readonly HISTFILESIZE
readonly HISTIGNORE
readonly HISTCONTROL
export HISTFILE HISTSIZE HISTFILESIZE HISTIGNORE HISTCONTROL
```

Damit dies funktioniert, dürfen die Nutzer nur Informationen zur `.bash_history`-Datei hinzufügen. Sie müssen daher *zusätzlich* die *append-only*-Option (nur-anfügen) mittels des Programms `chattr` für die `.bash_history` aller Nutzer setzen [17].

Beachten Sie, dass Sie obige Konfiguration auch in `.profile` des Benutzers eintragen können. Dann müssten Sie aber die Rechte korrekt vergeben, so dass der Benutzer daran gehindert ist, diese Datei zu verändern. Dies schließt ein, dass das Home-Verzeichnis der Benutzers diesem *nicht* gehört (sonst könnte er die Datei einfach löschen). Gleichzeitig müsste ihm ermöglicht werden, die Konfigurationsdatei `.profile` zu lesen und in `.bash_history` zu schreiben.

[16] Wenn HISTSIZE eine sehr große Zahl zugewiesen wird, kann dies bei einigen Shells zu Problemen führen, da der Verlauf für jede Sitzung eines Nutzers im Speicher abgelegt wird. Sie sind auf der sichereren Seite, wenn Sie HISTSIZE auf einen ausreichend großen Wert setzen und eine Kopie der History-Datei des Benutzers anlegen (falls Sie aus irgendwelchen Gründen den ganzen Verlauf von einem Nutzer benötigen).

[17] Ohne das Append-Only-Flag wäre es den Nutzern möglich, den Inhalt des Verlaufs zu löschen, indem sie `> .bash_history` ausführen.

Falls Sie diesen Weg gehen wollen, wäre es auch gut, das *immutable*-Flag (unveränderbar) für `.profile` zu setzen (auch dazu verwenden Sie `chattr`).

Vervollständigung der Nutzerüberwachung durch Accounting-Werkzeuge

Die vorherigen Beispiele sind ein einfacher Weg, um die Überwachung von Nutzern einzurichten. Sie eignen sich aber nicht unbedingt für komplexe Systeme oder für solche, auf denen die Nutzer überhaupt keine (oder ausschließlich) Shells am Laufen haben. Sollte dies der Fall sein, schauen Sie sich das Paket `acct` an, das Werkzeuge zur Bilanzierung (accounting utilities) enthält. Diese werden alle Kommandos, die ein Nutzer oder ein Prozess auf dem System ausführt, auf die Kosten von Plattenplatz aufzeichnen.

Wenn Sie diese Bilanzierung aktivieren, werden alle Informationen über Prozesse und Nutzer unter `/var/account/` gespeichert, genauer gesagt in `pacct`. Das Accounting-Paket schließt einige Werkzeuge (`sa`, `ac` und `lastcomm`) zur Analyse dieser Daten ein.

Andere Methoden zur Benutzerüberwachung

Wenn Sie wirklich paranoid sind und jedes Kommando des Nutzers protokollieren wollen, könnten Sie den Quellcode der `Bash` so ändern, dass sie alles, das der Nutzer eingibt in einer anderen Datei ablegt. Oder Sie lassen `ttysnoop` ununterbrochen jedes neue tty [18] überwachen und die Ausgaben in einer Datei speichern. Ein anderes nützliches Programm ist `snoopy` (vergleichen Sie auch the project page (`http://sourceforge.net/projects/snoopylogger/`)). Dies ist ein für den Nutzer transparentes Programm, das sich als eine Bibliothek einhängt und eine Hülle um *execve()*-Aufrufe bildet. Jedes ausgeführte Kommando wird im `syslogd` aufgezeichnet, indem die `authpriv`-Möglichkeit benutzt wird (üblicherweise wird dies unter `/var/log/auth.log` gespeichert).

4.11.10 Nachprüfung der Nutzerprofile

Wenn Sie *sehen* wollen, was Nutzer tatsächlich tun, wenn sie sich beim System anmelden, können Sie die `wtmp`-Datenbank benutzen, die alle Login-Informationen enthält. Diese Datei kann mit verschiedenen Werkzeugen weiterverarbeitet werden, unter ihnen `sac`, das ein Profil für jeden Nutzer ausgeben kann und zeigt, in welchem Zeitfenster sie sich für gewöhnlich auf dem System einloggen.

Für den Fall, dass Sie Accounting aktiviert haben, können Sie auch die mitgelieferten Werkzeuge verwenden, um festzustellen, wann Nutzer auf das System zugreifen und was sie ausführen.

[18] Ttys werden für lokal Logins und entfernte Logins mit ssh und telnet erzeugt.

4.11.11 umasks der Nutzer einstellen

Abhängig von Ihren Richtlinien möchten Sie vielleicht ändern, wie Nutzer Informationen teilen können. Dabei geht es um die Standardrechte von neu erstellten Dateien.

Wenn die Standardwerte von Debian für Ihr System zu großzügig sind, müssen Sie die umask-Einstellungen für alle Shells ändern. Strengere Umask-Einstellungen sind 027 (kein Zugriff der Gruppe *other* auf neue Dateien, dazu zählen andere Benutzer auf dem System) oder 077 (kein Zugriff der Mitglieder der Gruppe des Benutzers). Debian erzeugt (standardmäßig[19]) für jeden Benutzer eine eigene Gruppe, so dass das einzige Gruppenmitglied der Benutzer selbst ist. Daher ergibt sich zwischen 027 und 077 kein Unterschied, da die Benutzergruppe nur den Benutzer selbst enthält.

Dies ändern Sie, indem Sie eine passende `umask` für alle Benutzer einstellen. Dazu müssen Sie einen `umask`-Aufruf in den Konfigurationsdateien aller Shells einfügen: `/etc/profile` (wird von allen Shells beachtet, die kompatibel mit Bourne sind), `/etc/csh.cshrc`, `/etc/csh.login`, `/etc/zshrc` und wahrscheinlich noch ein paar andere (je nachdem, welche Shells Sie auf Ihrem System installiert haben). Sie können auch die *UMASK*-Einstellung in `/etc/login.defs` verändern. Von all diesen Dateien erlangt die letzte, die von der Shell geladen wird, Vorrang. Die Reihenfolge lautet: die Standard-System-Konfiguration für die Shell des Benutzers (d.h. `/etc/profile` und andere systemweite Konfigurationsdateien) und dann die Shell des Benutzers (seine `~/.profile`) und `~/.bash_profile` etc.). Allerdings können einige Shells mit dem *nologin*-Wert ausgeführt werden, was verhindern kann, dass einige dieser Dateien ausgewertet werden. Sehen Sie in der Handbuchseite Ihrer Shell für weitere Informationen nach.

Bei Anmeldungen, die von `login` Gebrauch machen, erhält die UMASK-Festlegung in `/etc/login.defs` Vorrang vor allen anderen Einstellungen. Dieser Wert wird aber nicht von Anwendungen des Benutzers beachtet, die nicht `login` verwenden, wie z.B. solche, die durch `su`, `cron` oder `ssh` ausgeführt werden.

Vergessen Sie nicht, die Dateien unter `/etc/skel/` zu überprüfen und gegebenenfalls anzupassen, da dort die Standards für Benutzer festgelegt werden, die mit dem Befehl `adduser` erstellt werden. Standardmäßig enthalten die Dateien in Debian keinen Aufruf von `umask`. Wenn sich aber ein solcher in Konfigurationsdateien befindet, sind neue Benutzer eher geneigt, ihn ihren Bedürfnissen anzupassen.

Beachten Sie allerdings, dass ein Nutzer seine `umask`-Einstellung abändern kann, wenn er es möchte, um sie großzügiger oder einschränkender zu machen, indem er seine Konfigurationsdateien verändert.

Das Paket `libpam-umask` passt die Standard-`umask` eines Benutzers mit Hilfe von PAM an. Nachdem Sie das Paket installiert haben, tragen Sie Folgendes in `/etc/pam.d/common-session` ein:

```
session    optional     pam_umask.so umask=077
```

[19] Wird in `/etc/adduser.conf` festgelegt (USERGROUPS=yes). Sie ändern dieses Verhalten, wenn Sie den Wert auf no setzen. Dies wird aber nicht empfohlen.

Zu guter Letzt sollte Sie in Betracht ziehen, die Standard-Umask von Root (022, wird in `/root/.bashrc` festgelegt) auf einen strengeren Wert zu verändern. Damit kann verhindert werden, dass der Systemadministrator als Root sensible Dateien in von allen lesbaren Verzeichnissen (wie z.B. `/tmp`) ablegt und sie so dem Durchschnittsbenutzer zugänglich macht.

4.11.12 Nutzer Sicht/Zugriff limitieren

FIXME: Inhalt benötigt. Aufzeigen der Folgen beim Upgraden, wenn die Paketrechte verändert werden, falls nicht `dpkg-statoverride` verwendet wird (übrigens sollte ein derartig paranoider Administrator seine Nutzer in eine `chroot`-Umgebung einsperren).

Wenn Sie einem Nutzer Zugriff auf das System mit einer Shell gewähren müssen, sollten Sie vorsichtig sein. Ein Nutzer kann normalerweise, wenn er sich nicht in einer streng abgeschirmten Umgebung befindet (z.B. in einem `chroot`-Gefängnis), ziemlich viel Informationen über Ihr System sammeln. Darunter fallen:

- einige Konfigurationsdateien unter `/etc`. Jedoch werden Debians Standardrechte für sensible Dateien (die zum Beispiel Passwörter enthalten könnten) den Zugriff auf kritische Informationen verhindern. Um zu sehen, auf welche Dateien nur der root Nutzer zugreifen kann, führen Sie zum Beispiel `find /etc -type f -a -perm 600 -a -uid 0` als Superuser aus.
- Ihre installierten Pakete. Indem man die Paket-Datenbank und das `/usr/share/doc`-Verzeichnis ansieht, oder indem man mit Hilfe der auf Ihrem System installierten Binaries und Bibliotheken versucht zu raten.
- einige Protokolle unter `/var/log`. Beachten Sie, dass auf einige Protokolle nur Root und die `adm`-Gruppe zugreifen kann (versuchen Sie `find /var/log -type f -a -perm 640`). Manche sind sogar ausschließlich für Root verfügbar (sehen Sie sich `find /var/log -type f -a -perm 600 -a -uid 0` an).

Was kann ein Nutzer von Ihrem System sehen? Wahrscheinlich ziemlich viele Sachen, versuchen Sie mal Folgendes (und jetzt tief durchatmen):

```
find / -type f -a -perm +006 2>/dev/null
find / -type d -a -perm +007 2>/dev/null
```

Was Sie sehen, ist eine Liste von allen Dateien, die ein Nutzer *einsehen* kann, und die Verzeichnisse, auf die er Zugriff hat.

Begrenzung des Zugangs zu Informationen anderer Nutzer

Wenn Sie immer noch Benutzern einen Shellzugang zur Verfügung stellen wollen, sollten Sie vielleicht die Informationen begrenzen, die man über anderen Nutzern einholen kann. Nutzer mit einer Shell haben die Neigung, eine ziemlich große Anzahl von Dateien in ihrem

$HOME zu erstellen: Mailboxen, persönliche Daten, Konfigurationen für X/GNOME/KDE-Anwendungen ...

In Debian wird jeder Nutzer mit einer zugehörigen Gruppe erstellt. Verschiedene Nutzer gehören dabei nie zur selben Gruppe. Folgendes ist das Standardverhalten: Wenn ein Benutzerkonto angelegt wird, wird auch eine Gruppe mit dem gleichen Namen erstellt. Dieser Gruppe wird der Nutzer zugewiesen. Damit wird die Idee einer allgemeinen *users*-Gruppe überflüssig, die es Nutzern erschweren könnte, Informationen vor anderen Nutzern zu verstecken.

Allerdings wird das *$HOME*-Verzeichnis der Benutzer mit 0755-Rechten (lesbar von der Gruppe, lesbar von der Welt) erstellt. Die Rechte Für die Gruppe sind kein Thema, da nur der Nutzer zu dieser Gruppe gehört. Allerdings könnten die Rechte für die Welt ein Problem darstellen, wobei dies von Ihren lokalen Grundsätzen abhängt.

Sie können dieses Verhalten so abändern, dass das Erstellen eines Nutzers andere Rechte für *$HOME* liefert. Um dieses Verhalten für *neue* Nutzer zu ändern, wenn sie erstellt werden, ändern Sie in der Konfigurationsdatei `/etc/adduser.conf` *DIR_MODE* auf 0750 (nicht lesbar für die Welt) ab.

Benutzer können immer noch Informationen austauschen, aber nicht mehr unmittelbar in ihrem *$HOME*-Verzeichnis, es sei denn, dass sie dessen Recht verändert haben.

Wenn Sie den Lesezugriff auf die Home-Verzeichnisse für die Welt verhindert, sollten Sie beachten, dass dann Nutzer ihre persönlichen Webseiten nicht unter `~/public_html` erstellen können, da der Webserver einen Teil des Pfads nicht lesen kann – und zwar das *$HOME*-Verzeichnis. Wenn Sie es Nutzern erlauben wollen, ihre HTML-Seiten in ihrem `~/public_html` zu veröffentlichen, sollten Sie *DIR_MODE* auf 0751 setzen. Das ermöglicht dem Webserver Zugang zum `public_html`-Verzeichnis (welches selber die Rechte 0755 haben sollte). So kann er den von den Nutzern veröffentlichten Inhalt anbieten. Natürlich sprechen wir hier nur über die Standardeinstellung. Benutzer können grundsätzlich die Rechte für ihre eigenen Dateien nach ihrem Gutdünken vergeben. Oder Sie können die Dinge, die für das Web bestimmt sind, in einem getrennten Ort ablegen, der kein Unterverzeichnis vom *$HOME*-Verzeichnis des Nutzers ist.

4.11.13 Erstellen von Benutzerpasswörtern

In vielen Fällen muss ein Administrator viele Benutzerkonten erstellen und alle mit Passwörtern ausstatten. Der Administrator könnte natürlich einfach als Passwort den Namen des Nutzerkontos vergeben. Dies wäre aber unter Sicherheitsgesichtspunkten nicht sehr klug. Ein besseres Vorgehen ist es, ein Programm zur Erzeugung von Passwörtern zu verwenden. Debian stellt die Pakete `makepasswd`, `apg` und `pwgen` zur Verfügung, die Programme liefern (deren Name ist der gleiche wie der des Pakets), die zu diesem Zweck verwendet werden können. `Makepasswd` erzeugt wirklich zufällige Passwörter, gibt also der Sicherheit gegenüber der Aussprechbarkeit den Vorzug. Dagegen versucht `pwgen`, bedeutungslose, aber aussprechbare Passwörter herzustellen (dies hängt natürlich auch von Ihrer Muttersprache ab). `Apg` liefert Algorithmen für beide Möglichkeiten (Es gibt auch eine Client/Server-Version dieses Programms. Diese befindet sich aber nicht im Debian-Paket).

`Passwd` erlaubt nur die interaktive Zuweisung von Passwörtern (da es direkt den tty-Zugang benutzt). Wenn Sie Passwörter ändern wollen, wenn Sie eine große Anzahl von Benutzern erstellen, können Sie diese unter der Verwendung von `adduser` mit der `--disabled-login`-Option erstellen, und danach `usermod` oder `chpasswd` [20] benutzen (beide Programme stammen aus dem `passwd`-Paket. Sie haben sie also schon installiert). Wenn Sie lieber eine Datei verwenden, die alle Informationen zur Erstellung von Nutzern als Batch-Prozess enthält, sind Sie vielleicht mit `newusers` besser dran.

4.11.14 Kontrolle der Benutzerpasswörter

Die Passwörter der Nutzer sind manchmal die *schwächste Stelle* der Sicherheit eines Systems. Das liegt daran, dass manche Nutzer schwache Passwörter für ihr Konto wählen (und je mehr Nutzer Zugang zum System haben, umso größer die Chance, dass das passiert). Selbst wenn Sie Überprüfungen mit dem PAM-Module cracklib und Grenzen für Passwörter einsetzen, wie in 'Nutzerauthentifizierung: PAM' auf Seite 58 beschrieben wird, ist es Nutzern immer noch möglich, schwache Passwörter zu verwenden. Da der Zugang der Nutzer auch den Zugang aus der Ferne (hoffentlich über `ssh`) umfassen kann, ist es wichtig, dass das Erraten von Passwörtern für entfernte Angreifer so schwierig wie möglich ist. Dies gilt insbesondere dann, wenn es ihnen gelungen sein sollte, Zugang zu wichtigen Informationen wie den Benutzernamen oder sogar den Dateien `passwd` und `shadow` selbst zu bekommen.

Ein Systemadministrator muss bei einer großen Anzahl von Nutzern überprüfen, ob deren Passwörter mit den lokalen Sicherheitsmaßstäben in Einklang stehen. Und wie überprüft man das? Indem man versucht, sie wie ein Angreifer zu knacken, der Zugriff auf die gehashten Passwörter hat (also auf die Datei `/etc/shadow`).

Ein Administrator kann `john` oder `crack` (beide benutzen Brute-Force (Rohe Gewalt) zum Knacken von Passwörtern) zusammen mit einer passenden Wörterliste verwenden, um die Passwörter der Nutzer zu überprüfen, und falls ein schlechtes Passwort entdeckt wird, geeignete Schritte unternehmen. Sie können mit `apt-cache search wordlist` nach Debian/GNU-Paketen suchen, die Wörterlisten enthalten, oder Sie besuchen die klassischen Internetseiten mit Wörterlisten wie `ftp://ftp.ox.ac.uk/pub/wordlists` oder `ftp://ftp.cerias.purdue.edu/pub/dict`.

4.11.15 Ausloggen von untätigen Nutzern

Untätige (idle) Nutzer stellen für gewöhnlich ein Sicherheitsproblem dar. Ein Nutzer kann untätig sein, da er Mittagessen ist, oder weil eine entfernte Verbindung hängen blieb und nicht wieder hergestellt wurde. Unabhängig von den Gründen können untätige Benutzer zu einer Kompromittierung führen:

- weil die Konsole des Benutzers vielleicht nicht verriegelt ist und damit ein Eindringling darauf zugreifen kann.

[20] `Chpasswd` kann keine MD5-Passwörter erzeugen. Daher muss ihm das Passwort in verschlüsselter Form übergeben werden, bevor es mit der `-e`-Option verwendet werden kann.

- weil ein Angreifer an eine schon beendete Netzwerkverbindung anknüpfen und Befehle an die entfernte Shell schicken kann (das ist ziemlich einfach, wenn die entfernte Shell, wie bei `telnet`, nicht verschlüsselt ist).

In einige entfernte System wurde sogar schon durch ein untätiges (und abgelöstes) `screen` eingedrungen.

Die automatische Trennung von untätigen Benutzern ist gewöhnlich ein Teil der lokalen Sicherheitsregeln, die durchgesetzt werden müssen. Es gibt mehrere Wege, dies zu tun:

- Wenn die Shell des Benutzers die `Bash` ist, kann ein Systemadministrator `TMOUT` einen Standardwert zuweisen (vergleich `bash(1)`). Das hat zur Folge, dass die Shell automatisch entferne, untätige Nutzer ausloggt. Beachten Sie, dass der Wert mit der `-o`-Option gesetzt werden muss. Ansonsten wäre es den Nutzern möglich, ihn zu verändern (oder zu löschen).

  Installieren Sie `timeoutd` und konfigurieren Sie `/etc/timeouts` passend zu Ihren lokalen Sicherheitsrichtlinien. Der Daemon achtet auch untätige Nutzer und beendet ihre Shells gegebenenfalls.

- Installieren Sie `autolog` und richten Sie es so ein, dass es untätige Nutzer entfernt.

Vorzugswürdige Methoden sind die Daemonen `timeoutd` und `autolog`, da letzten Endes die Nutzer ihre Standardshell ändern können oder zu einer anderen (unbeschränkten) Shell wechseln können, nachdem sie ihre Standardshell gestartet haben.

4.12 Die Nutzung von tcpwrappers

TCP-Wrapper (Schutzumschläge für TCP) wurden entwickelt, als es noch keine echten Paketfilter gab, aber Zugangskontrollen notwendig waren. Trotzdem sind sie immer noch hoch interessant und nützlich. Ein TCP-Wrapper erlaubt Ihnen, einem Host oder einer Domain einen Dienst anzubieten oder zu verweigern, und standardmäßig Zugriff zu erlauben oder zu verweigern (das alles wird auf der Anwendungsebene durchgeführt). Wenn Sie mehr Informationen haben möchten, sehen Sie sich `hosts_access(5)` an.

Viele der unter Debian installierten Dienste

- werden entweder durch den TCP-Wrapper Service (`tcpd`) aufgerufen,
- oder wurden mit libwrapper (Bibliothek für TCP-Wrapper) Unterstützung kompiliert.

Einerseits werden Sie bei manchen Diensten (einschließlich `telnet`, `ftp`, `netbios`, `swat` und `finger`), die in `/etc/inetd.conf` konfiguriert werden, sehen, dass die Konfigurationsdatei zuerst `/usr/sbin/tcpd` aufruft. Andererseits, selbst wenn ein Dienst nicht über den `inetd`-Superdaemon ausgeführt wird, kann die Unterstützung von Tcp-Wrapper einkompiliert werden. Dienste, die unter Debian mit TCP-Wrappern kompiliert wurden, sind

`ssh`, `portmap`, `in.talk`, `rpc.statd`, `rpc.mountd`, `gdm`, `oaf` (der GNOME-Aktivierungs-Daemon), `nessus` und viele andere.

Um herauszufinden, welche Pakete TCP-Wrapper benutzen[21], versuchen Sie Folgendes:

```
$ apt-cache rdepends libwrap0
```

Beachten Sie bitte Folgendes, wenn Sie `tcpchk` (ein sehr nützliches Programm zur Überprüfung der TCP-Wrapper-Konfiguration und -Syntax) laufen lassen. Wenn Sie Stand-Alone-Dienste (alleinstehende Dienste, also solche, die direkt mit der Wrapper-Bibliothek verbunden sind) der `host.deny`- oder `host.allow`-Datei hinzufügen, wird `tcpchk` Sie warnen, dass er sie nicht finden kann, da er sie nur in `/etc/inetd.conf` sucht (die Handbuchseite ist an dieser Stelle nicht sehr genau).

Jetzt kommt ein kleiner Trick und vielleicht die kleinste Alarmanlage zur Erkennung von Eindringlingen: Im Allgemeinen sollten Sie eine anständige Firewall als erste und TCP-Wrapper als zweite Verteidigungslinie haben. Der Trick besteht nun darin, ein *SPAWN*-Kommando [22] in `/etc/hosts.deny` einzutragen, das immer dann eine Mail an Root schickt, wenn ein Dienst abgewiesen wurde:

```
ALL: ALL: SPAWN ( \
  echo -e "\n\
  TCP Wrappers\: Verbindungsaufbau abgelehnt\n\
  By\: $(uname -n)\n\
  Prozess\: %d (pid %p)\n\
  Nutzer\: %u\n\
  Host\: %c\n\
  Datum\: $(date)\n\
" | /usr/bin/mail -s "Verbindung zu %d blockiert" root) &
```

Achtung: Das obige Beispiel kann sehr leicht zu DoS (Denial of Service, Verbindungsaufbau abgelehnt) führen, indem man versucht, sehr viele Verbindungen in kurzer Zeit aufzubauen. Viele E-Mails bedeuten viel Dateiaktivität, die lediglich durch das Senden von ein paar Paketen erreicht wird.

4.13 Die Wichtigkeit von Logs und Alarmen

Es ist leicht einzusehen, dass die Behandlung von Logs (Protokolldateien) und Alarmen eine wichtige Angelegenheit in einem sicheren System ist. Stellen Sie sich vor, ein System ist perfekt

[21] Bei älteren Veröffentlichungen von Debian sollte Sie Folgendes ausführen:

```
$ apt-cache showpkg libwrap0 | egrep '^[[:space:]]' | sort -u | \ sed
's/,libwrap0$//;s/^[[:space:]]\+//'
```

[22] Beachten Sie hier die Schreibweise, da *spawn* nicht funktionieren wird.

konfiguriert und zu 99% sicher. Wenn ein Angriff unter dieses 1% fällt, und es keine Sicherheitsmaßnahmen gibt, dies erstens zu erkennen und zweitens einen Alarm auszulösen, so ist das System überhaupt nicht sicher.

Debian GNU/Linux stellt Werkzeuge zur Verfügung, die die Analyse von Log-Dateien übernehmen. Am beachtenswertesten sind `swatch` [23], `logcheck` oder `loganalysis` (alle Pakete werden ein wenig Anpassung benötigen, um unnötige Dinge aus den Reports zu entfernen). Wenn sich das System in Ihrer Nähe befindet, könnte es nützlich sein, das System-Log auf einer virtuellen Konsole auszugeben. Die ist nützlich, da Sie so (auch von weiter weg oder im Vorbeigehen) sehen können, ob sich das System richtig verhält. Debians `/etc/syslog.conf` wird mit einer auskommentierten Standardkonfiguration ausgeliefert. Um diese Ausgabe einzuschalten, entfernen Sie die Kommentarzeichen vor den entsprechenden Zeilen und starten `syslog` neu (`/etc/init.d/syslogd restart`):

```
daemon,mail.*;\
      news.=crit;news.=err;news.=notice;\
      *.=debug;*.=info;\
      *.=notice;*.=warn       /dev/tty8
```

Um die Logs farbig zu gestalten sollten einen Blick auf `colorize`, `ccze` oder `glark` werfen. Es gibt da noch eine Menge über die Analyse von Logs zusagen, das hier nicht behandelt werden kann. Eine gute Quelle für weiter Informationen ist die Webseite Log Analysis (http://www.loganalysis.org/). In jedem Fall sind selbst automatische Werkzeuge dem besten Analysewerkzeug nicht gewachsen: Ihrem Gehirn.

4.13.1 Nutzung und Anpassung von `logcheck`

Das Paket `logcheck` ist in Debian auf drei Pakete verteilt: `logcheck` (das Hauptprogramm), `logcheck-database` (eine Datenbank regulärer Ausdrücke für das Programm) und `logtail` (gibt Logzeilen aus, die noch nicht gelesen wurden). Der Standard unter Debian (in `/etc/cron.d/logcheck`) ist, dass `logcheck` jede Stunde und nach jedem Neustart ausgeführt wird.

Wenn dieses Werkzeug in geeigneter Weise angepasst wurde, kann es sehr nützlich sein, um den Administrator zu alarmieren, wenn etwas ungewöhnliches auf dem System passiert. `Logcheck` kann vollständig angepasst werden, so dass es Mails über Ereignisse aus den Logs sendet, die Ihrer Aufmerksamkeit bedürfen. Die Standard-Installation umfasst Profile zum ignorieren von Ereignissen und Rechtswidrigkeiten für drei unterschiedliche Setups (Workstation, Server und paranoid). Das Debian-Paket umfasst die Konfigurationsdatei `/etc/logcheck/logcheck.conf`, die vom Programm eingelesen wird, und die definiert, an welchen Nutzer die Testergebnisse geschickt werden sollen. Es stellt außerdem einen Weg für Pakete zur Verfügung, um neue Regeln in folgenden Verzeichnisses zu erstellen: `/etc/logcheck/cracking.d`

[23] Es gibt darüber einen ziemlich guten Artikel von Lance Spitzner (http://www.spitzner.net/swatch.html).

/_packagename_ /etc/logcheck/violations.d/_packagename_, /etc/logcheck/violations.ignore.d/_packagename_, /etc/logcheck/ignore.d.paranoid/_packagename_, /etc/logcheck/ignore.d.server/_packagename_, und /etc/logcheck/ignore.d.workstation/_packagename_. Leider benutzen das noch nicht viele Pakete. Wenn Sie ein Regelwerk entwickelt haben, dass für andere Nutzer nützlich sein könnte, schicken Sie bitte einen Fehlerbericht für das entsprechende Paket (als ein *wishlist*-Fehler). Mehr Informationen finden Sie unter /usr/share/doc/logcheck/README.Debian.

logcheck konfiguriert man am besten, indem man nach der Installation seine Hauptkonfigurationsdatei /etc/logcheck/logcheck.conf bearbeitet. Verändern Sie den Benutzer, an den die Berichte geschickt werden (standardmäßig ist das Root). Außerdem sollten Sie auch den Schwellenwert für Berichte festlegen. logcheck-database hat drei Schwellenwerte mit steigender Ausführlichkeit: Workstation (Arbeitsplatz), Server und paranoid. "server" ist der Standardwert, "paranoid" wird nur für Hochsicherheitsmaschinen empfohlen, auf denen so wenig Dienste wie möglich laufen. "workstation" eignet sich für relativ geschützte, nicht kritische Maschinen. Wenn Sie neue Log-Dateien hinzufügen wollen, müssen Sie diese nur zu /etc/logcheck/logcheck.logfiles hinzufügen. Es ist für die standardmäßige Sysloginstallation eingerichtet.

Wenn Sie dies geschafft haben, sollten Sie die nächsten Tage/Wochen/Monate die verschickten Mails überprüfen. Falls Sie Nachrichten finden, die Sie nicht erhalten wollen, fügen Sie die regulären Ausdrücke (regular expressions, vergleiche regex(7) und egrep(1)), die zu diesen Nachrichten passen, in /etc/logcheck/ignore.d.*reportlevel*/local ein. Versuchen Sie, dass der reguläre Ausdruck mit der gesamten Logzeile übereinstimmt. Details, wie man Regeln schreibt, finden Sie in /usr/share/doc/logcheck-database/README.logcheck-database.gz. Das ist ein andauernder Prozess der Abstimmung. Wenn nur noch relevante Meldungen verschickt werden, können Sie davon ausgehen, dass dieser Prozess beendet ist. Beachten Sie, dass logcheck, selbst wenn er läuft, Ihnen keine Mail schickt, wenn er nichts Relevantes auf Ihrem System findet (so bekommen Sie höchstens eine Mail pro Woche, wenn Sie Glück haben).

4.13.2 Konfiguration, wohin Alarmmeldungen geschickt werden

Debian wird mit einer Standardkonfiguration für Syslog (in /etc/syslog.conf) ausgeliefert, so dass Meldungen je nach System in die passenden Dateien geschrieben werden. Das sollte Ihnen bereits bekannt sein. Falls nicht, werfen Sie einen Blick auf die Datei syslog.conf und deren Dokumentation. Wenn Sie ein sicheres System betreuen wollen, sollten Ihnen bekannt sein, wohin Log-Meldungen geschickt werden, so dass sie nicht unbeachtet bleiben.

Zum Beispiel ist es für viele Produktiv-Systeme sinnvoll, Meldungen auch auf der Konsole auszugeben. Aber bei vielen solcher Systeme ist es sehr wichtig, auch eine neue Maschine zu haben, die für die anderen als ein Loghost fungiert (d.h. sie empfängt die Logs aller anderen Systeme).

Sie sollten auch an Mails für Root denken, da viele Sicherheits-Kontrollen (wie snort) ihre Alarme an die Mailbox von root senden. Diese Mailbox zeigt normalerweise auf den ersten

Nutzer, der auf dem System erstellt wurde (prüfen Sie dazu `/etc/aliases`). Sorgen Sie dafür, dass roots Mails irgendwo hin geschickt werden, wo sie auch gelesen werden (lokal oder ferngesteuert).

Es gibt noch andere Accounts mit besonderer Funktion und andere Aliase auf Ihrem System. Auf einem kleinen System ist es wohl am einfachsten, sicherzustellen, dass alle Aliase auf den Root-Account zeigen, und dass Mails an root in das persönliche Postfach des System-Administrator weiter geleitet werden.

FIXME: It would be interesting to tell how a Debian system can send/receive SNMP traps related to security problems (jfs). Check: `snmptrapfmt`, `snmp` and `snmpd`.

4.13.3 Nutzen eines loghosts

Ein Loghost ist ein Host der die syslog-Daten über ein Netzwerk sammelt. Wenn eine Ihrer Maschinen geknackt wird, kann der Eindringling seine Spuren nicht verwischen, solange er den Loghost nicht ebenfalls geknackt hat. Demzufolge muss der Loghost also besonders sicher sein. Aus einer Maschinen einen Loghost zu machen ist relativ einfach: Starten Sie den `syslogd` einfach mit `syslogd -r`, und ein neuer Loghost ist geboren. Um dies unter Debian dauerhaft zu machen, editieren Sie `/etc/init.d/sysklogd` und ändern Sie die Zeile

```
SYSLOGD=""
```

in

```
SYSLOGD="-r"
```

Als nächstes konfigurieren Sie die anderen Maschinen, Ihre Daten an den Loghost zu senden. Fügen Sie einen Eintrag, ähnlich dem folgenden zu der `/etc/syslog.conf` hinzu:

```
facility.level          @Ihr_Loghost
```

Schauen Sie in die Dokumentation um zu erfahren, wodurch Sie *facility* und *level* ersetzen können (Sie sollten nicht wörtlich übernommen werden). Wenn Sie alles fern mit loggen wollen, schreiben Sie einfach:

```
*.*                     @Ihr_Loghost
```

in Ihre `syslog.conf`. Sowohl lokal als auch entfernt mitzuloggen ist die beste Lösung (ein Angreifer könnte davon ausgehen, dass er seine Spuren verwischt hat, nachdem er die lokale Log-Datei gelöscht hat). Für weitere Informationen sehen Sie sich die Handbuch-Seiten `syslog(3)`, `syslogd(8)` und `syslog.conf(5)` an.

4.13.4 Zugriffsrechte auf Log-Dateien

Es ist nicht nur wichtig zu entscheiden, wie Warnungen genutzt werden, sondern auch, wer hierauf Zugriff hat, d.h. wer Log-Dateien (falls Sie nicht einen Log-Host verwenden) lesen oder verändern kann. Sicherheits-Alarme, die ein Angreifer verändern oder abschalten kann, sind im Falle eines Eindringens nicht viel wert. Außerdem sollten Sie berücksichtigen, dass Log-Dateien einem Eindringling ziemlich viel Informationen über Ihr System verraten, wenn er auf sie Zugriff hat.

Einige Zugriffsrechte auf Log-Dateien sind nach der Installation nicht gerade perfekt (aber das hängt natürlich von Ihren lokalen Sicherheitsmaßstäben ab). Zuerst einmal müssen `/var/log` `/lastlog` und `/var/log/faillog` nicht für normale Nutzer lesbar sein. In der `lastlog`-Datei können Sie sehen, wer sich zuletzt eingeloggt hat. In `faillog` befindet sich eine Zusammenfassung fehlgeschlagener Logins. Der Autor empfiehlt, die Rechte von beiden auf 660 zu setzen (mit `chmod 660`). Werfen Sie einen kurzen Blick auf Ihre Log-Dateien, und entscheiden Sie sehr vorsichtig, welche Log-Dateien Sie les- oder schreibbar für einen Nutzer mit einer anderen UID als 0 und einer anderen Gruppe als 'adm' oder 'root' machen. Sie können dies sehr leicht auf Ihrem System überprüfen:

```
#  find /var/log -type f -exec ls -l {} \; | cut -c 17-35 |sort -u
(see to what users do files in /var/log belong)
#  find /var/log -type f -exec ls -l {} \; | cut -c 26-34 |sort -u
(see to what groups do files in /var/log belong)
# find /var/log -perm +004
(files which are readable by any user)
#  find /var/log \! -group root \! -group adm -exec ls -ld {} \;
(files which belong to groups not root or adm)
```

Um anzupassen, wie neue Log-Dateien erstellt werden, müssen Sie wahrscheinlich das Programm anpassen, das sie erstellt. Wenn die Log-Dateien rotiert werden, können Sie das Verhalten der Erstellung und Rotation anpassen.

4.14 Den Kernel patchen

Debian GNU/Linux stellt verschiedene Patches für den Linux-Kernel zur Verfügung, die die Sicherheit erhöhen:

- Erkennung von Eindringlingen für Linux (Linux Intrusion Detection (`http://www.lids.org`), enthalten im Paket `lids-2.2.19`). Dieser Kernelpatch erleichtert Ihnen, Ihr Linuxsystem abzuhärten, indem er Ihnen ermöglicht, Prozesse einzuschränken, zu verstecken und zu schützten, sogar vor Root. Er führt Fähigkeiten für eine zwingende Zugangskontrolle ein.
- Linux Trustees (`http://trustees.sourceforge.net/`) (im Paket `trustees`). Dieser Patch fügt ein ordentliches, fortgeschrittenes Rechtemanagement Ihrem Linux-Kernel

hinzu. Besondere Objekte, die "trustees" (Treuhänder) genannt werden, sind mit jeder Datei oder Verzeichnis verbunden. Sie werden im Speicher des Kernels abgelegt und erlauben so eine schnelle Abfrage aller Rechte.

- NSA Enhanced Linux (im Paket `selinux`. Backports von Paketen, die SElinux unterstützen, sind unter http://selinux.alioth.debian.org/ erhältlich. Weiterführende Informationen können Sie auf der SElinux in Debian Wiki Seite (http://wiki.debian.org/SELinux) und auf Manoj Srivastavas (http://www.golden-gryphon.com/software/security/selinux.xhtml) und Russell Cookers' (http://www.coker.com.au/selinux/) SElinux-Webseiten finden.

- Der Exec-Shield-Patch (http://people.redhat.com/mingo/exec-shield/) aus dem Paket `kernel-patch-exec-shield`. Dieser Patch schützt vor einigen Pufferüberläufen (stack smashing attacks).

- Der Grsecurity-Patch (http://www.grsecurity.net/) aus den Paketen `kernel-patch-2.4-grsecurity` und `kernel-patch-grsecurity2` [24] verwirklicht Mandatory Access Control durch RBAC und stellt Schutz vor Pufferüberläufen durch PaX, ACLs, Zufälligkeit im Netzwerk (um die Erkennung von Spuren des OS zu erschweren) und noch viele Features mehr (http://www.grsecurity.net/features.php) zur Verfügung.

- `kernel-patch-adamantix` bietet die Patches an, die für die Debian-Distribution Adamantix (http://www.adamantix.org/) entwickelt wurden. Dieser Patch für den Kernel 2.4.x führt einige Sicherheitsfähigkeiten wie nichtausführbaren Speicher durch den Einsatz von PaX (http://pageexec.virtualave.net/) und Mandatory Access Control auf Grundlage von RSBAC (http://www.rsbac.org/) ein. Andere Features sind der Random-PID-Patch (http://www.vanheusden.com/Linux/sp/), ein mit AES verschlüsseltes Loop-Gerät, Unterstützung von MPPE und eine Zurückportierung von IPSEC v2.6.

- `cryptoloop-source`. Dieser Patch erlaubt Ihnen, die Fähigkeiten der Crypto-API des Kernels zu verwenden, um verschlüsselte Dateisysteme mit dem Loopback-Gerät zu erstellen.

- Kernel-Unterstützung von IPSEC (im Paket `kernel-patch-freeswan`). Wenn Sie das IPsec-Protokoll mit Linux verwenden wollen, brauchen Sie diesen Patch. Damit können

[24] Beachten Sie, dass ein Konflikt zwischen diesem Patch und den Patches besteht, die schon im Quellpaket des Kernels 2.4 von Debian enthalten sind. Sie werden den Vanilla-Kernel verwenden müssen. Dazu führen Sie folgende Schritte durch:

```
# apt-get install kernel-source-2.4.22 kernel-patch-debian-2.4.22 # tar
xjf /usr/src/kernel-source-2.4.22.tar.bz2 # cd kernel-source-2.4.22 #
/usr/src/kernel-patches/all/2.4.22/unpatch/debian
```

Für weitere Informationen siehe #194225 (http://bugs.debian.org/194225), #199519 (http://bugs.debian.org/199519), #206458 (http://bugs.debian.org/206458), #203759 (http://bugs.debian.org/203759), #204424 (http://bugs.debian.org/204424), #210762 (http://bugs.debian.org/210762), #211213 (http://bugs.debian.org/211213) und die Diskussion auf debian-devel (http://lists.debian.org/debian-devel/2003/debian-devel-200309/msg01133.html).

Sie ziemlich leicht VPNs erstellen, sogar mit Windows-Rechnern, da IPsec ein verbreiteter Standard ist. IPsec-Fähigkeiten wurden in den Entwicklungskernel 2.5 eingefügt, so dass dieses Feature standardmäßig im zukünftigen Kernel 2.6 enthalten sein wird. Homepage: http://www.freeswan.org. Hinweis: Die Verwendung von FreeSwan wurde zugunsten von OpenSwan aufgegeben. *FIXME*: Der neuste Kernel 2.4 in Debian enthält eine Rückeinbindung des IPSEC-Codes aus 2.5. Kommentar dazu.

Die folgenden Sicherheitspatches für den Kernel sind nur noch für alte Kernelversionen in Woody verfügbar und werden nicht mehr weiterentwickelt:

- POSIX Access Control Lists (http://acl.bestbits.at/) (ACLs, Listen zur Zugangskontrolle) für Linux im Paket `kernel-patch-acl`. Dieser Kernelpatch stellt Listen zur Zugangskontrolle zur Verfügung. Das ist eine fortgeschrittene Methode, um den Zugang zu Dateien einzuschränken. Es ermöglicht Ihnen, den Zugang zu Dateien und Verzeichnisses fein abzustimmen.
- Der Patch für den Linux-Kernel Openwall (http://www.openwall.com/linux/) von Solar Designer, der im Paket `kernel-patch-2.2.18-openwall` enthalten ist. Er enthält eine nützliche Anzahl von Beschränkungen des Kernels wie eingeschränkte Verweise, FIFOs in `/tmp`, ein begrenztes `/proc`-Dateisystem, besondere Handhabung von Dateideskriptoren, einen nichtausführbaren Bereich des Stapelspeichers des Nutzers und andere Fähigkeiten. Hinweis: Dieser Patch ist nur auf die Kernelversion 2.2 anwendbar, für 2.4 werden von Solar keine Pakete angeboten.
- `kernel-patch-int`. Auch dieser Patch fügt kryptographische Fähigkeiten zum Linux-Kernel hinzu. Er war bis zu den Debian-Releases bis Potato nützlich. Er funktioniert nicht mehr mit Woody. Falls Sie Sarge oder eine neuere Version verwenden, sollten Sie einen aktuelleren Kernel einsetzen, in dem diese Features bereits enthalten sind.

Wie auch immer, einige Patches werden von Debian noch nicht zur Verfügung gestellt. Wenn Sie denken, dass manche von ihnen hinzugefügt werden sollten, fragen Sie danach auf Arbeitbedürfende und voraussichtliche Pakete (http://www.de.debian.org/devel/wnpp/).

4.15 Schutz vor Pufferüberläufen

Pufferüberlauf (buffer overflow) wird eine verbreitete Art von Angriffen auf Software [25] genannt, die die unzureichende Überprüfung von Eingabegrenzen ausnutzen (ein Programmierfehler, der häufig bei der Programmiersprache C auftritt), um durch Programmeingaben Befehle auf der Maschine auszuführen. Diese Attacken über Server, die auf Verbindungen warten, oder über lokal installierte Software, die einem Nutzer größere Privilegien gewährt (`setuid` oder `setgid`) kann zu einem kompromittierten System führen.

Es gibt hauptsächlich vier Methoden, um sich gegen Pufferüberläufe zu schützen:

[25] Sie sind in der Tat so verbreitet, dass sie die Grundlage für 20% aller gemeldeten Sicherheitsmängel pro Jahr darstellen, wie von statistics from ICAT's vulnerability database (http://icat.nist.gov/icat.cfm?function=statistics) herausgefunden wurde.

- Patchen Sie den Kernel, um das Ausführen des Stapelspeichers zu verhindern. Sie können entweder Exec-Shield, OpenWall oder PaX (ist in den Grsecurity- und Adamantix-patches enthalten) verwenden.

- Benutzen Sie eine Bibliothek wie libsafe (http://www.research.avayalabs.com/project/libsafe/), um verwundbare Funktionen zu überschreiben und ordentliche Prüfungen einzuführen (Informationen wie man `libsafe` installiert finden Sie hier (http://www.Linux-Sec.net/harden/libsafe.uhow2.txt)).

- Verbessern Sie den Quellcode, indem Sie Werkzeuge einsetzen, die Teile finden, die zu dieser Verwundbarkeit führen könnten.

- Übersetzen Sie den Quellcode neu, um vernünftige Prüfungen einzuführen, um Überläufe zu verhindern. Benutzen Sie dazu den Stack Smashing Protector (SSP) (http://www.research.ibm.com/trl/projects/security/ssp/) Patch für GCC (der von Adamantix (http://www.adamantix.org) verwendet wird).

Debian GNU/Linux liefert bis einschließlich der Release 3.0 Software, um alle dieser Methoden bis auf den Schutz bei der Übersetzung des Quellcodes (das wurde aber schon in Fehler #213994 (http://bugs.debian.org/213994) nachgefragt) zu implementieren.

Beachten Sie, dass selbst wenn Debian einen Compiler zur Verfügung stellen würde, der Schutz vor Stapel- und Pufferüberläufen bieten würde, so doch alle Pakete neu übersetzt werden müssten, um diese Eigenschaft einzuführen. Tatsächlich ist das die Aufgabe der Distribution Adamantix (unter anderen Fähigkeiten). Die Auswirkungen dieses neuen Features auf die Stabilität der Software muss aber noch ermittelt werden (einige Programme und einige Prozessoren werden vielleicht deswegen nicht mehr funktionieren).

Seien Sie auf jeden Fall gewarnt, dass selbst diese Umgehungen des Problems nicht vor Pufferüberläufen schützen können, da es Möglichkeiten gibt, diese zu überlisten, wie in Ausgabe 58 (http://packetstorm.linuxsecurity.com/mag/phrack/phrack58.tar.gz) des phrack-Magazins oder in COREs Advisory Multiple vulnerabilities in stack smashing protection technologies (http://online.securityfocus.com/archive/1/269246) beschrieben.

Wenn Sie Ihren Schutz gegen Pufferüberläufe (unabhängig von der gewählten Methode) testen wollen, können Sie `paxtest` installieren und die angebotenen Tests laufen lassen.

4.15.1 Kernelpatch zum Schutz vor Pufferüberläufen

Ein Kernelpatch, der Schutz vor Pufferüberläufen bietet, ist der Openwall-Patch, der diese im Linux-Kernel 2.2 verhindern soll. Für 2.4 oder neuere Kernel müssen Sie die Umsetzung von Exec-Shield oder die von PaX (ist im Grsecurity-Patch `kernel-patch-2.4-grsecurity` und im Adamantix-Patch `kernel-patch-adamantix` enthalten) benutzen. Für weitere Informationen zum Einsatz dieser Patches lesen Sie den Abschnitt ‘Den Kernel patchen’ auf Seite 78.

4.15.2 Schutz durch `libsafe`

Es ist ziemlich einfach, ein Debian GNU/Linux-System mit `libsafe` zu schützen. Sie müssen nur das Paket installieren und *Ja* sagen, damit die Bibliothek global geladen wird. Seien Sie dennoch vorsichtig, da das Software unbrauchbar machen kann (besonders Programme, die mit der alten `libc5` verknüpft sind). Sie sollten also zuerst die gemeldeten Fehlerberichte (`http://bugs.debian.org/libsafe`) lesen und die kritischsten Programme auf Ihrem System mit dem Programm zum Einhüllen von `libsafe` testen.

Wichtiger Hinweis: Der Schutz durch `libsafe` könnte im Moment nicht wirkungsvoll sein, wie es in 173227 (`http://bugs.debian.org/173227`) beschrieben wird. Testen Sie es gründlich, bevor Sie es in einer produktiven Umgebung einsetzen, und verlassen Sie sich nicht ausschließlich darauf, um Ihr System zu schützen.

4.15.3 Prüfprogramme für Pufferüberläufe

Zur Nutzung von Werkzeugen zum Aufspüren von Pufferüberläufen benötigen Sie in jedem Fall Programmiererfahrung, um den Quellcode zu reparieren (und neu zu kompilieren). Debian stellt beispielsweise `bfbtester` (einen Überlauftester, der Programme per Brute-Force (durch Testen aller Möglichkeiten) nach Überläufen der Kommandozeile und von Umgebungsvariablen durchtestet) bereit. Andere interessante Pakete sind auch `rats`, `pscan`, `flawfinder` und `splint`.

4.16 Sichere Übertragung von Dateien

Während der normalen Systemadministration müssen Sie immer mal wieder Dateien auf Ihr System spielen oder von diesem holen. Auf sichere Art und Weise Dateien von einem Host zu einem anderen zu kopieren, wird durch die Benutzung des Paketes `ssh` erreicht. Eine andere Möglichkeit ist die Nutzung von `ftpd-ssl`, einem ftp-Server der *Secure Socket Layer* benutzt, um Übertragungen zu verschlüsseln.

Jede dieser Methoden benötigt natürlich einen speziellen Client. Debian stellt Ihnen solche zur Verfügung, zum Beispiel enthält das Paket `ssh` das Programm `scp`. Es arbeitet wie `rcp`, aber ist komplett verschlüsselt, so dass die *bösen Jungs* noch nicht einmal herausbekommen können, WAS Sie kopieren. Wie es den Server gibt, so gibt es natürlich auch ein `ftp-ssl` Client-Paket. Sie können Clients für diese Software sogar für andere (nicht-UNIXoide) Betriebssysteme finden. `putty` und `winscp` stellen eine secure-copy-Implementierung für jede Version von Microsoft-Betriebssystemen zur Verfügung.

Beachten Sie, dass die Verwendung von `scp` den Nutzern Zugang zum gesamten Dateisystem ermöglicht, es sei denn, dass es in eine `chroot`-Umgebung eingesperrt ist, wie es in ‘SSH in ein Chroot-Gefängnis einsperren’ auf Seite 100 beschrieben wird. Wahrscheinlich sogar leichter (abhängig vom verwendeten Daemon) kann auch der FTP in eine `chroot`-Umgebung eingesperrt werden. Das wird in ‘Absichern von FTP’ auf Seite 102 beschrieben. Falls Sie sich sorgen, dass Nutzer Ihre lokalen Dateien durchsehen, und Sie verschlüsselte Kommunikation

wünschen, können Sie einen FTP-Daemon mit Unterstützung für SSL einrichten oder FTP mit Klartext und VPN verbinden (siehe 'Virtual Private Networks (virtuelle private Netzwerke)' auf Seite 165.

4.17 Einschränkung und Kontrolle des Dateisystems

4.17.1 Benutzung von Quotas

Es ist wichtig, eine gute Quota-Regelung zu haben, da es die Nutzer daran hindert, die Festplatten zu füllen.

Sie können zwei Arten von Quota-Systemen benutzen: Nutzer-Quota und Gruppen-Quota. Wie Sie sicher denken können, limitiert User-Quota den Plattenplatz, den ein Nutzer belegen kann, und Gruppen-Quota macht dasselbe für ganze Gruppen. Beachten Sie dies, wenn Sie die Größe der Quotas festlegen.

Es gibt ein paar wichtige Punkte, über die Sie nachdenken sollten, wenn Sie ein Quota-System aufsetzen:

- Halten Sie die Quotas klein genug, so dass die Nutzer Ihre Festplatte nicht aufzehren können.
- Halten Sie die Quotas groß genug, so dass Nutzer sich nicht beschweren oder dass Ihr Mail-Quota Sie daran hindert, nach einer Weile Mails anzunehmen.
- Nutzen Sie Quotas auf allen Bereichen, die Nutzer beschreiben können, auf `/home` ebenso wie auf `/tmp`.

Für jede Partition und jedes Verzeichnis, auf das Nutzer Schreibzugriff haben, sollte ein Quota eingerichtet werden. Berechnen Sie eine sinnvolle Quota-Größe, die Benutzerfreundlichkeit und Sicherheit kombiniert, und weisen Sie diese zu.

So, nun wollen Sie Quotas benutzen. Zuerst müssen Sie prüfen, ob Ihr Kernel Quota unterstützt. Wenn nicht, müssen Sie ihn neu kompilieren. Prüfen Sie anschließend, ob das Paket `quota` installiert ist. Wenn nicht, installieren Sie es.

Um Quota für die entsprechenden Dateisysteme einzuschalten, müssen Sie nur die Einstellung `defaults` in Ihrer `/etc/fstab` zu `defaults,usrquota` ändern. Wenn Sie Gruppen-Quotas benötigen, ersetzen Sie `usrquota` durch `grpquota`. Sie können auch beides verwenden. Erstellen Sie dann leere `quota.user` und `quota.group` in den Hauptverzeichnissen der Dateisysteme, auf denen Sie Quotas einführen möchten (d.h. `touch /home/quota.user /home/quota.group` für das Dateisystem `/home`).

Starten Sie `quota` neu, indem Sie `/etc/init.d/quota stop;/etc/init.d/quota start` ausführen. Nun sollte quota laufen, und die Größen können festgelegt werden.

Bearbeiten der Quotas eines bestimmten Nutzer wird mit `edquota -u <user>` gemacht. Gruppen-Quotas können mit `edquota -g <group>` geändert werden. Setzen Sie dann die weiche und die harte Grenze und inode-Quotas, falls Sie es benötigen.

Mehr Informationen über Quotas finden Sie im Handbuch von quot und im Mini-Howto von quota (`/usr/share/doc/HOWTO/de-html/mini/DE-Quota-HOWTO.html`). Sie sollten auch einen Blick auf `pam_limits.so` werfen.

4.17.2 Die für das ext2-Dateisystem spezifischen Attribute (chattr/lsattr)

Zusätzlich zu den normalen Unix-Rechten bieten die ext2- und ext3-Dateisysteme eine Anzahl von besonderen Attributen, die Ihnen mehr Kontrolle über die Dateien auf Ihrem System erlauben. Im Gegensatz zu den gewöhnlichen Rechten werden diese Attribute nicht vom gebräuchlichen Befehl `ls -l` angezeigt und können auch nicht mit `chmod` geändert werden. Um sie zu verwalten, brauchen Sie zwei weitere Programme, nämlich `lsattr` und `chattr` (im Paket `e2fsprogs`). Beachten Sie, dass das bedeutet, dass diese Attribute normalerweise bei einem Backup des Systems nicht gespeichert werden. Wenn Sie also eines verändern, könnte es sich lohnen, die aufeinander folgenden `chattr`-Befehle in einem Skript zu speichern, damit Sie sie später wieder zuweisen können, falls Sie ein Backup zurückspielen müssen.

Unter allen Attributen werden die zwei, die für die Erhöhung der Sicherheit am bedeutendsten sind, mit den Buchstaben 'i' und 'a' bezeichnet. Sie können nur vom Superuser vergeben (oder entfernt) werden:

- Das Attribut 'i' ('immutable', unveränderlich): Eine Datei mit diesem Attribut kann weder verändert noch gelöscht oder umbenannt werden, nicht einmal vom Superuser. Auch ein Link auf sie kann nicht angelegt werden.
- Das Attribut 'a' ('append', anfügen): Dieses Attribut hat den gleichen Effekt für das Attribut immutable, allerdings mit der Ausnahme, dass Sie immer noch die Datei im Anfügen-Modus öffnen können. Das bedeutet, dass Sie ihr immer noch Inhalt hinzufügen, aber den vorhanden Inhalt nicht verändern können. Dieses Attribut ist besonders für die Protokolldateien nützlich, die unter `/var/log/` gespeichert werden. Beachten Sie aber, dass sie durch Log-Rotations-Skripte manchmal verschoben werden.

Diese Attribute können auch für Verzeichnisse vergeben werden. In diesem Fall ist es jedem unmöglich gemacht, den Inhalt des Verzeichnisses zu verändern, also beispielsweise eine Datei umzubenennen oder zu löschen. Wenn das append-Attribut einem Verzeichnis zugewiesen wird, können nur noch Dateien erstellt werden.

Es ist leicht einzusehen, wie das 'a' Attribut die Sicherheit verbessert, indem es Programmen, die nicht vom Superuser ausgeführt werden, die Fähigkeit einräumt, Daten hinzuzufügen, aber verhindert, dass älterer Inhalt verändert wird. Dem gegenüber erscheint das 'i' Attribut uninteressanter. Schließlich kann der Superuser ja schon die normalen Unix-Rechte verwenden, um den Zugang zu Dateien einzuschränken. Und ein Angreifer, der Zugang zum Konto des Superusers hat, könnte immer das Programm `chattr` benutzen, um die Attribute zu entfernen. Ein solcher Eindringlich ist vielleicht zunächst verwirrt, wenn er feststellt, dass er eine

Datei nicht löschen kann. Aber Sie sollten nicht davon ausgehen, dass er blind ist – immerhin hat er es geschafft, in Ihr System einzudringen! Einige Handbücher (einschließlich früherer Versionen dieses Dokuments) empfehlen, einfach die Programme `chattr` und `lsattr` vom System zu entfernen, um die Sicherheit zu erhöhen. Aber diese Strategie, die auch als "security by obscurity" (Sicherheit durch Verschleierung) bekannt ist, sollte unter allen Umständen vermieden werden, da sie ein falsches Gefühl von Sicherheit vermittelt.

Dieses Problem lösen Sie auf sichere Art und Weise, indem Sie die Fähigkeiten des Linux-Kernel verwenden, wie es in 'Aktive Verteidigung' auf Seite 190 beschrieben wird. Die hier interessante Fähigkeit heißt `CAP_LINUX_IMMUTABLE`: Wenn Sie es vom Satz der Fähigkeiten entfernen (indem Sie zum Beispiel den Befehl `lcap CAP_LINUX_IMMUTABLE` verwenden, ist es nicht mehr möglich, irgendwelche 'a' oder 'i' Attribute auf Ihrem System zu verändern, auch nicht dem Superuser! Ein vollständige Strategie könnte also folgendermaßen aussehen:

1 Vergeben Sie die Attribute 'a' und 'i' an von Ihnen gewünschte Dateien.

2 Fügen Sie den Befehl `lcap CAP_LINUX_IMMUTABLE` einem der Skripten, die den Start des Systems steuern (startup scripts), hinzu.

3 Setzen Sie das Attribut 'i' für dieses Skript, andere Startdateien und auch das Programm `lcap` selbst.

4 Führen Sie den oben genannten Befehl per Hand aus (oder starten Sie Ihr System neu, um sicherzustellen, dass alles wie gewünscht funktioniert).

4.17.3 Prüfung der Integrität des Dateisystems

Sind Sie sich sicher, dass `/bin/login` auf Ihrer Festplatte immer noch dasselbe Programm ist, das Sie vor ein paar Monaten installiert haben? Was wäre, wenn es sich um eine gehackte Version handelt, die eingegebene Passwörter in einer versteckten Datei ablegt oder sie als Klartext im ganzen Internet herummailt?

Die einzige Methode einen gewissen Schutz dafür zu haben ist es, die Dateien jede(n) Stunde/Tag/Monat (ich ziehe täglich vor) zu prüfen, indem man deren aktuelle und alte MD5-Summe vergleicht. Zwei unterschiedliche Dateien können keine gleichen MD5-Summen haben (die MD5-Summe umfasst 128 Bits, so ist die Wahrscheinlichkeit, dass zwei unterschiedliche Dateien eine gleiche MD5-Summe haben etwa 1 zu 3,4e3803). So sind Sie sicher, solange niemand den Algorithmus gehackt hat, der die MD5-Summen auf Ihrer Maschine erstellt. Dies ist, nun ja, extrem schwer und sehr unwahrscheinlich. Sie sollten diese Überprüfung Ihrer Programme als sehr wichtig ansehen.

Weit verbreitete Tools hierfür sind `sxid`, `aide` (Advanced Intrusion Detection Environment, fortgeschrittene Umgebung zur Erkennung von Eindringlingen), `tripwire`, `integrit` und `samhain`. Das Installieren von `debsums` wird Ihnen helfen, die Integrität des Dateisystems zu überprüfen, indem Sie die MD5-Summen jeder Datei gegen die MD5-Summe aus dem Debian-Archiv-Paket vergleichen. Seien Sie aber gewarnt, dass diese Dateien sehr leicht von einem Angreifer geändert werden können. Außerdem stellen nicht alle Pakete MD5-Summen für die

in ihnen enthaltenen Programme zur Verfügung. Weitere Informationen finden Sie unter 'Periodische Überprüfung der Integrität' auf Seite 187 und 'Einen Schnappschuss des Systems erstellen' auf Seite 94.

Sie benutzen vielleicht `locate`, um das gesamte Dateisystem zu indizieren. Wenn das so ist, sollten Sie die Auswirkungen davon berücksichtigen. Das Debianpaket `findutils` enthält `locate`, das als Nutzer nobody läuft. Daher indiziert es nur Dateien, die von jedermann eingesehen werden können. Wenn Sie dieses Verhalten verändern, werden allerdings alle Orte von Dateien für alle Nutzer sichtbar. Wenn Sie das gesamte Dateisystem indizieren wollen (und nicht nur die Stückchen, die der Nutzer nobody sehen kann), können Sie `locate` durch das Paket `slocate` ersetzen. slocate wird als eine um Sicherheit erweiterte Version von GNU locate bezeichnet, hat aber tatsächlich weitere Funktionen zum Auffinden von Dateien. Wenn Sie `slocate` benutzen, sieht ein Benutzer nur Dateien, auf die er auch Zugriff hat, während Sie Dateien und Verzeichnisse des gesamten Systems ausschließen können. Das Paket `slocate` führt seinen Aktualisierungsprozess mit höheren Rechten aus als locate. Außerdem indiziert es jede Datei. Nutzern wird es dadurch ermöglicht, schnell nach jeder Datei zu suchen, die sie sehen können. `slocate` zeigt ihnen keine neuen Dateien an; es filtert die Ausgabe auf Grundlage der UID.

Sie sollten auch `bsign` oder `elfsign` einsetzen. `elfsign` bietet die Möglichkeit, digitale Signaturen an ELF-Binaries anzufügen und diese Signaturen zu überprüfen. Die aktuelle Fassung verwendet PKI, um die Checksummen der Binaries zu signieren. Dies hat den Vorteil, dass festgestellt werden kann, ob das Binary verändert wurde und wer es erstellt hat. `bsign` verwendet GPG, `elfsign` benutzt PKI-(X.509)-Zertifikate (OpenSSL).

4.17.4 Aufsetzen einer Überprüfung von setuid

Das Debian-Paket `checksecurity` enthält einen `Cron`-Job, der täglich in `/etc/cron.daily/checksecurity` ausgeführt wird. [26]. Dieser `Cron`-Job führt das Skript `/usr/sbin/checksecurity` aus, das Informationen über Änderungen sichert.

Das Standard-Verhalten sendet diese Informationen nicht an den Superuser. Stattdessen erstellt es eine tägliche Kopie dieser Änderungen unter `/var/log/setuid.changes`. Sie sollten die Variable MAILTO (in `/etc/checksecurity.conf`) auf 'root' setzen, damit diese Informationen an ihn gemailt werden. Sehen Sie sich auch `checksecurity(8)` für weitere Konfigurations-Informationen an.

4.18 Absicherung des Netzwerkzugangs

FIXME: Mehr (für Debian spezifischer) Inhalt benötigt.

[26] In älteren Veröffentlichungen war checksecurity in cron integriert und die Datei hieß `/etc/cron.daily/standard`.

4.18.1 Konfiguration der Netzwerkfähigkeiten des Kernels

Viele Fähigkeiten des Kernels können während des Betriebs verändert werden, indem etwas an das `/proc`-Dateisystem geschickt wird, oder indem `sysctl` benutzt wird. Wenn Sie `/sbin/sysctl -A` eingeben, können Sie sehen, was Sie einstellen können und was die Optionen sind. Veränderungen werden vorgenommen, indem `/sbin/sysctl -w variable=value` ausgeführt wird (vergleiche `sysctl(8)`). Nur in seltenen Fällen müssen Sie hier etwas bearbeiten. Aber auch hier können Sie die Sicherheit erhöhen. Zum Beispiel:

```
net/ipv4/icmp_echo_ignore_broadcasts = 1
```

Dies ist ein *Windows Emulator*, weil es sich wie Windows bei Rundrufen (Broadcast-Ping) verhält, wenn es auf 1 gesetzt wird. Das bedeutet, dass ICMP-Echo-Anfragen, die an die Rundrufadresse geschickt werden, ignoriert werden. Anderenfalls macht es gar nichts.

Falls Sie verhindern wollen, dass Ihr System auf ICMP-Echo-Anfragen antwortet, müssen Sie nur diese Konfigurationsoption anschalten:

```
net/ipv4/icmp_echo_ignore_all = 1
```

Verwenden Sie Folgendes, um Pakete mit unmöglichen Adressen (erzeugt durch falsche Routen) in Ihrem Netzwerk zu protokollieren:

```
/proc/sys/net/ipv4/conf/all/log_martians = 1
```

Für weiterführende Informationen dazu, welche Sachen mit `/proc/sys/net/ipv4/*` angestellt werden können, sollten Sie `/usr/src/linux/Documentation/filesystems/proc.txt` lesen. Alle Optionen werden gründlich in `/usr/src/linux/Documentation/networking/ip-sysctl.txt` [27] beschrieben.

4.18.2 Konfiguration von Syncookies

Diese Option ist ein zweischneidiges Schwert. Auf der einen Seite schützt es Ihr System vor dem Überfluten mit syn-Paketen. Auf der anderen Seite verletzt es definierte Standards (RFCs).

```
net/ipv4/tcp_syncookies = 1
```

Wenn Sie das dauerhaft für den Kernel festlegen wollen, müssen Sie in `/etc/network/options` `syncookies=yes` festlegen. Jedes Mal, wenn `/etc/init.d/networking` ausgeführt wird (was typischerweise beim Booten geschieht), wird diese Option wirksam. Dagegen wird folgendes nur eine einmalige Wirkung bis zum nächsten Neustart haben:

[27] In Debian kopiert das Paket `kernel-source-`*`version`* die Kernelquellen nach `/usr/src/kernel-source-`*`version`*`.tar.bz2`. Ersetzen Sie einfach *version* mit der installierten Kernelversion.

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

Diese Option ist nur verfügbar, wenn der Kernel mit `CONFIG_SYNCOOKIES` übersetzt wurde. Alle Kernel von Debian wurden mit dieser Option kompiliert. Sie können das folgendermaßen überprüfen:

```
$ sysctl -A |grep syncookies
net/ipv4/tcp_syncookies = 1
```

Weitere Informationen zu TCP-Syncookies finden Sie unter http://cr.yp.to/syncookies.html.

4.18.3 Absicherung des Netzwerks beim Hochfahren

Wenn Sie die Netzwerkoptionen des Kernels konfigurieren, müssen Sie dafür sorgen, dass sie bei jedem Neustart des Systems geladen werden. Das nachfolgende Beispiel aktiviert neben vielen der oben vorgestellten Optionen auch noch ein paar andere nützliche Optionen.

Tatsächlich gibt es zwei Möglichkeiten, Ihr Netzwerk beim Booten einzurichten. Sie können entweder `/etc/sysctl.conf` konfigurieren (siehe `sysctl.conf(5)`) oder ein Skript einsetzen, das beim Aktivieren der Netzwerkschnittstellen aufgerufen wird. Die erste Möglichkeit wird auf alle Schnittstellen angewendet, die zweite erlaubt es Ihnen, die Konfiguration für jede Schnittstelle separat zu wählen.

Ein Beispiel einer Konfiguration von `/etc/sysctl.conf`, die einige Netzwerkoptionen auf der Kernelebene absichert, wird unten gezeigt. Beachten Sie darin den Kommentar, dass `/etc/network/options` beim Ausführen von `/etc/init.d/networking` (dies ist in der Startsequenz nach `procps`) einige Werte überschreiben könnte, wenn sich Werte in dieser Datei widersprechen.

```
#
# /etc/sysctl.conf - Configuration file for setting system variables
# See sysctl.conf (5) for information. Also see the files under
# Documentation/sysctl/, Documentation/filesystems/proc.txt, and
# Documentation/networking/ip-sysctl.txt in the kernel sources
# (/usr/src/kernel-$version if you have a kernel-package installed)
# for more information of the values that can be defined here.

#
# Be warned that /etc/init.d/procps is executed to set the following
# variables.  However, after that, /etc/init.d/networking sets some
# network options with builtin values.  These values may be overridden
# using /etc/network/options.
#
#kernel.domainname = example.com
```

```
# Additional settings - adapted from the script contributed
# by Dariusz Puchala (see below)
# Ignore ICMP broadcasts
net/ipv4/icmp_echo_ignore_broadcasts = 1
#
# Ignore bogus ICMP errors
net/ipv4/icmp_ignore_bogus_error_responses = 1
#
# Do not accept ICMP redirects (prevent MITM attacks)
net/ipv4/conf/all/accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
# net/ipv4/conf/all/secure_redirects = 1
#
# Do not send ICMP redirects (we are not a router)
net/ipv4/conf/all/send_redirects = 0
#
# Do not forward IP packets (we are not a router)
# Note: Make sure that /etc/network/options has 'ip_forward=no'
net/ipv4/conf/all/forwarding = 0
#
# Enable TCP Syn Cookies
# Note: Make sure that /etc/network/options has 'syncookies=yes'
net/ipv4/tcp_syncookies = 1
#
# Log Martian Packets
net/ipv4/conf/all/log_martians = 1
#
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
# Note: Make sure that /etc/network/options has 'spoofprotect=yes'
net/ipv4/conf/all/rp_filter = 1
#
# Do not accept IP source route packets (we are not a router)
net/ipv4/conf/all/accept_source_route = 0
```

Um dieses Skript verwenden zu können, müssen Sie es zuerst unter z.B. `/etc/network/interface-secure` (der Name ist nur ein Beispiel) erstellen und es wie folgt aus `/etc/network/interfaces` aufrufen:

```
auto eth0
iface eth0 inet static
      address xxx.xxx.xxx.xxx
```

```
        netmask 255.255.255.xxx
        broadcast xxx.xxx.xxx.xxx
        gateway xxx.xxx.xxx.xxx
        pre-up /etc/network/interface-secure
```

In diesem Beispiel wird das Skript aufgerufen, um alle Netzwerkschnittstellen abzusichern, wie unten gezeigt wird, bevor die Schnittstelle eth0 aktiviert wird.

```
#!/bin/sh -e
# Skriptname: /etc/network/interface-secure
#
# Verändert das Standardverhalten für alle Schnittstellen in einigen Bereiche
# um vor TCP/IP-Spoofing und Angriffen zu schützen
#
# Wurde von Dariusz Puchalak beigesteuert
#
# Broadcast echo protection enabled
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# IP forwarding disabled
echo 0 > /proc/sys/net/ipv4/conf/all/forwarding

# TCP syn cookies protection enabled
echo 1 > /proc/sys/net/ipv4/tcp_syncookies

# Log strange packets (this includes spoofed packets, source routed packets,
# redirect packets) but be careful with this on heavy loaded web servers
echo 1 >/proc/sys/net/ipv4/conf/all/log_martians

# Bad error message protection enabled
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses

# IP spoofing protection
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

# Disable ICMP redirect acceptance
echo 0 > /proc/sys/net/ipv4/conf/all/accept_redirects
echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects

# Disable source routed packets
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route

exit 0
```

Beachten Sie, dass Sie auch verschiedene Netzwerkoptionen für verschiedene Schnittstellen (falls Sie mehr als eine haben) haben können, indem Sie die pre-up-Zeile verändern:

```
pre-up /etc/network/interface-secure $IFACE
```

Zusätzlich müssen Sie ein Skript verwenden, das Änderungen nur auf eine bestimmte Schnittstelle anwendet und nicht auf alle Schnittstellen. Beachten Sie aber, dass einige Netzwerkoptionen nur global gesetzt werden können. Dies ist ein Beispielsskript:

```
#!/bin/sh -e
# Skriptname: /etc/network/interface-secure
#
# Verändert das Standardverhalten für alle Schnittstellen in einigen Bereiche
# um vor TCP/IP-Spoofing und Angriffen zu schützen
#
# Wurde von Dariusz Puchalak beigesteuert

IFACE=$1
if [ -z "$IFACE" ] ; then
   echo "$0: Must give an interface name as argument!"
   echo "Usage: $0 <interface>"
   exit 1
fi

if [ ! -e /proc/sys/net/ipv4/conf/$IFACE/ ]; then
   echo "$0: Interface $IFACE does not exit (cannot find /proc/sys/net/ipv4/c
   exit 1
fi

# IP forwarding disabled
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/forwarding

# Log strange packets (this includes spoofed packets, source routed packets,
# redirect packets) but be careful with this on heavy loaded web servers
echo 1 >/proc/sys/net/ipv4/conf/$IFACE/log_martians

# IP spoofing protection
echo 1 > /proc/sys/net/ipv4/conf/$IFACE/rp_filter

# Disable ICMP redirect acceptance
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/accept_redirects
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/send_redirects

# Disable source routed packets
echo 0 > /proc/sys/net/ipv4/conf/$IFACE/accept_source_route

exit 0
```

Eine andere Lösungsmöglichkeit ist es, ein `init.d`-Skript zu erstellen und es beim Booten

auszuführen (verwenden Sie `update-rc.d`, um die passenden `rc.d`-Links herzustellen).

4.18.4 Konfiguration der Firewall

Um die Möglichkeiten einer Firewall zu haben, damit entweder das lokale System oder andere *dahinter* beschützt werden, muss der Kernel mit Firewall-Unterstützung kompiliert worden sein. Der Standardkernel von Debian 2.2 (Linux 2.2) stellt die Paketfilter-Firewall `ipchains` zur Verfügung. Der Standardkernel von Debian 3.0 (Linux 2.4) enthält die *stateful* Paketfilter-Firewall `iptables` (netfilter).

In jedem Fall ist es recht einfach, einen anderen als den mit Debian gelieferten Kernel zu benutzen. Sie finden vorkompilierte Kernel als Pakete vor, die Sie leicht auf Ihrem Debian-System installieren können. Mit Hilfe des Pakets `kernel-source-`*`X`* können Sie auch die Kernelquellen herunterladen und einen maßgeschneiderten Kernel kompilieren, indem Sie `make-kpkg` aus dem Paket `kernel-package` benutzen.

Auf das Aufsetzen einer Firewall unter Debian wird unter ‘Hinzufügen von Firewall-Fähigkeiten’ auf Seite 122 ausführlich eingegangen.

4.18.5 Lösung des Problems der Weak-End-Hosts

Auf Systemen mit mehr als einer Schnittstelle zu verschiedenen Netzwerken können Dienste so eingerichtet werden, dass sie Verbindungen nur zu einer bestimmte IP-Adresse zulassen. Normalerweise verhindert das Zugang zu diesen Diensten, wenn an sie Anfragen über andere Adressen gestellt werden. Allerdings bedeutet das nicht, dass der Dienst an eine bestimmte *Hardware*-Adresse (Netzwerkkarte) gebunden ist (ein verbreiteter Irrtum). [28]

Das ist kein Problem von ARP und auch keine Verletzung eines RFCs (es wird in RFC1122 (`ftp://ftp.isi.edu/in-notes/rfc1122.txt`), Abschnitt 3.3.4.2 als *weak end host* bezeichnet). Vergessen Sie nicht, dass IP-Adressen nichts mit dem physischen Schnittstellen zu tun haben.

Im Kernel 2.2 (und davor) könnte dieses Problem so gelöst werden:

```
# echo 1 > /proc/sys/net/ipv4/conf/all/hidden
# echo 1 > /proc/sys/net/ipv4/conf/eth0/hidden
# echo 1 > /proc/sys/net/ipv4/conf/eth1/hidden
.....
```

[28] Um das nachzuvollziehen folgendes Beispiel, das von Felix von Leitner auf der Bugtraq-Mailingliste vorgestellt wurde:

```
host a (eth0 connected to eth0 of host b): ifconfig eth0 10.0.0.1 ifconfig eth1
23.0.0.1 tcpserver -RHl localhost 23.0.0.1 8000 echo fnord host b: ifconfig eth0
10.0.0.2 route add 23.0.0.1 gw 10.0.0.1 telnet 23.0.0.1 8000
```

Das scheint allerdings nicht mit Diensten zu funktionieren, die mit 127.0.0.1 verbunden sind. Sie sollten vielleicht für die Tests raw sockets verwenden.

Bei späteren Kernel kann das folgendermaßen gelöst werden:

- Regeln für iptables.
- Richtig konfiguriertes Routing. [29] Oder:
- Patchen des Kernels. [30]

In diesem Text finden sich viele Fälle, in denen gezeigt wird, wie man einige Dienste (sshd-Server, apache, Druckserver, ...) so konfiguriert, dass sie nur auf einer bestimmten Adresse lauschen. Der Leser sollte in Betracht ziehen, dass das den Zugang aus dem gleichen (lokalen) Netzwerk nicht verhindern kann, wenn nicht die in diesem Abschnitt vorgeschlagenen Schritte ergriffen werden. [31]

FIXME: Comments on Bugtraq indicate there is a Linux specific method to bind to a given interface.

FIXME: Submit a bug against netbase so that the routing fix is standard behavior in Debian?

4.18.6 Schutz vor ARP-Angriffen

Wenn Sie den anderen Kisten in Ihrem LAN nicht trauen (das sollte immer so sein, da es die sicherste Einstellung ist), sollten Sie sich vor den verschiedenen ARP-Angriffen schützen.

Wie Sie wissen, wird das ARP-Protokoll dazu verwendet, IP-Adressen mit MAC-Adressen zu verknüpfen (für alle Details siehe RFC826 (`ftp://ftp.isi.edu/in-notes/rfc826.txt`)). Jedes Mal, wenn Sie ein Paket an eine IP-Adresse schicken, wird eine ARP-Auflösung vorgenommen (zuerst wird in den lokalen ARP-Speicher geschaut, und falls die IP nicht im Speicher ist, wird ein Rundruf (Broadcast) mit der ARP-Anfrage verschickt), um die Hardware-Adresse des Ziels zu finden. Alle ARP-Angriffe zielen darauf ab, Ihrem Rechner vorzugaukeln, dass die IP-Adresse des Rechners B mit der MAC-Adresse des Computers des Angreifers verbunden ist. Dadurch wird jedes Paket, das Sie an den Rechner B, der mit der IP-Adresse verbunden ist, schicken wollen, an den Computer des Eindringlings umgeleitet ...

Diese Angriffe (Verfälschung des ARP-Speichers, ARP-Spoofing, ...) ermöglichen dem Angreifer, auf Netzwerken den Verkehr abzuhören (selbst bei Netzwerken, die über einen Switch

[29] Die Tatsache, dass dieses Verhalten durch Routing geändert werden kann, wurde von Matthew G. Marsh in dem Bugtraq-Thread beschrieben:

```
eth0 = 1.1.1.1/24 eth1 = 2.2.2.2/24 ip rule add from 1.1.1.1/32 dev lo table 1 prio
15000 ip rule add from 2.2.2.2/32 dev lo table 2 prio 16000 ip route add default dev
eth0 table 1 ip route add default dev eth1 table 2
```

[30] Wie im Bugtraq-Thread beschrieben, gibt es dafür einige Patches auf `http://www.linuxvirtualserver.org/~julian/#hidden` und `http://www.fefe.de/linux-eth-forwarding.diff`.

[31] Ein Angreifer, der nicht in der gleichen Broadcast-Domain (also dem gleichen Netzwerk) wie der angegriffene Host ist, kann auf viele Probleme bei Zugang stoßen, nachdem die Anbindung der IP-Adressen konfiguriert wurde. Wenn der Angriff über einen Router läuft, kann es sich als ziemlich schwer herausstellen, die Antworten zurückzubekommen.

laufen). Er kann sich leicht in eine Verbindung einschleusen oder einen Host vom Netzwerk nehmen oder ... ARP-Angriffe sind leistungsfähig und einfach durchzuführen. Es gibt dafür auch einige Werkzeuge wie `arpspoof` aus dem Paket `dsniff` oder arpoison (http://arpoison.sourceforge.net/).

Allerdings gibt es immer eine Lösung:

- Verwenden Sie einen statischen ARP-Speicher. So erstellen Sie "statische" Einträge in Ihrem ARP-Speicher:

  ```
  arp -s host_name hdwr_addr
  ```

  Indem Sie statische Einträge für jeden wichtigen Host in Ihrem Netzwerk vergeben, stellen Sie sicher, dass niemand einen (falschen) Eintrag für diese Hosts erstellen oder verändern kann (statische Einträge verfallen nicht und können nicht verändert werden). Auch gefälschte ARP-Antworten werden ignoriert.

- Entdecken Sie verdächtigen ARP-Verkehr. Sie können dazu `arpwatch`, `karpski` oder allgemeinere IDS, die auch verdächtigen ARP-Verkehr entdecken können wie `snort` oder prelude (http://www.prelude-ids.org), einsetzen.

- Verwirklichen Sie einen IP-Filter, der die MAC-Adressen überprüft.

4.19 Einen Schnappschuss des Systems erstellen

Bevor Sie das System in eine produktive Umgebung stellen, können Sie einen Schnappschuss des gesamten Systems machen. Diesen Schnappschuss können Sie im Falle einer Kompromittierung (siehe 'Nach einer Kompromittierung (Reaktion auf einem Vorfall)' auf Seite 195) benutzen. Sie sollten so einen Schnappschuss immer dann erneuern, wenn Sie das System aktualisieren, insbesondere wenn Sie auf eine neue Debian Release upgraden.

Hierfür können Sie beschreibbare, austauschbare Datenträger benutzen, die Sie schreibschützen können. Dies kann eine Diskette (die nach der Benutzung schreibgeschützt wird), eine CD in einem CD-ROM-Laufwerk (Sie können auch wiederbeschreibbare CD-ROMs benutzen, so können Sie sogar alte Sicherheitskopien Ihrer MD5-Summen behalten), eine USB-Platte oder eine MMC-Karte (wenn Ihr System auf diese zugreifen kann und sie schreibgeschützt werden können) sein.

Das folgende Skript erstellt einen solchen Schnappschuss:

```
#!/bin/bash
/bin/mount /dev/fd0 /mnt/floppy
if [ ! -f /usr/bin/md5sum ] ; then
 echo "Kann nicht md5sum finden. Breche ab."
 exit 1
fi
```

```
/bin/cp /usr/bin/md5sum /mnt/floppy
echo "Erstelle MD5-Datenbank"
>/mnt/floppy/md5checksums.txt
for dir in /bin/ /sbin/ /usr/bin/ /usr/sbin/ /lib/ /usr/lib/
do
   find $dir -type f | xargs /usr/bin/md5sum >>/mnt/floppy/md5checksums-lib.t
done
echo "MD5-Datenbank (nach der Installation) erstellt"
if [ ! -f /usr/bin/sha1sum ] ; then
 echo "Kann nicht sha1sum finden"
else
 /bin/cp /usr/bin/sha1sum /mnt/floppy
 echo "Erstelle SHA-1-Datenbank"
 >/mnt/floppy/sha1checksums.txt
 for dir in /bin/ /sbin/ /usr/bin/ /usr/sbin/ /lib/ /usr/lib/
 do
    find $dir -type f | xargs /usr/bin/sha1sum >>/mnt/floppy/sha1checksums-li
 done
 echo "SHA-1-Datenbank (nach der Installation) erstellt"
fi
/bin/umount /dev/fd0
exit 0
```

Beachten Sie, dass das Programm md5sum (und sha1sum, falls verfügbar) auch auf der Diskette gesichert werden muss, so dass Sie es später benutzen können, um die anderen Programme Ihres Systems zu prüfen (für den Fall, dass md5sum oder sha1sum einen Trojaner enthalten). Wenn Sie aber sicher sein wollen, dass Sie eine gültige Kopie von md5sum verwenden, sollten Sie eine statische Kopie von md5sum erstellen und diese verwenden (damit wird verhindert, dass eine manipulierte libc-Bibliothek das Programm beeinträchtigt) oder md5sum nur in einer sauberen Umgebung einsetzen, die Sie etwa mit einer Rettungs-CD-ROM oder einer Live-CD erzeugen können (damit wird verhindert, dass ein manipulierter Kernel das Programm beeinflusst). Ich kann es nicht genug betonen: Wenn Sie ein System haben, in das eingebrochen wurde, können Sie den Ausgaben nicht vertrauen. Sehen Sie sich auch 'Nach einer Kompromittierung (Reaktion auf einem Vorfall)' auf Seite 195 an.

Dieser Schnappschuss enthält nicht die Dateien unterhalb von `/var/lib/dpkg/info`, wo MD5-Summen installierter Pakete enthalten sind (die Dateien enden mit `.md5sums`). Sie können diese Informationen zusätzlich kopieren, aber Sie sollten Folgendes beachten:

- die Dateien mit den MD5-Summen enthalten die MD5-Summen aller Dateien, die ein Debian-Paket enthält, nicht nur die der Systemprogramme. Das hat zur Folge, dass diese Datenbank viel größer ist (5 MB statt 600 KB auf einem Debian GNU/Linux System mit graphischen Subsystem und etwa 2,5 GB Software installiert) und nicht auf ein kleines, transportables Medium wie eine Diskette passt, aber wohl auf einen tragbaren USB-Speicher.

- nicht alle Debian-Pakete stellen MD5-Summen der installierten Dateien zur Verfügung, da es (derzeit) nicht in der Policy verlangt wird. Sie können allerdings nach der Installation die MD5-Summen aller Pakete mit `debsums` erstellen:

```
# debsums --generate=missing,keep
```

Sobald der Schnappschuss erstellt wurde, sollten Sie sicherstellen, dass das entsprechende Medium schreibgeschützt ist. Sie können es dann als Sicherheitskopie verwenden oder in ein Laufwerk stecken, um jede Nacht mit `cron` die MD5-Summen des Systems mit Ihrem Schnappschuss zu vergleichen.

Wenn Sie keine Überprüfung von Hand einrichten wollen, können Sie immer eines der Integritätssysteme verwenden, die diese Aufgabe und noch vieles mehr für Sie erledigen werden. Weitere Informationen finden Sie unter 'Periodische Überprüfung der Integrität' auf Seite 187.

4.20 Andere Empfehlungen

4.20.1 Benutzen Sie keine Software, die von svgalib abhängt

SVGAlib ist ganz nett für Konsolen-Liebhaber wie mich, aber in der Vergangenheit wurde mehrfach gezeigt, dass es ziemlich unsicher ist. Exploits durch `zgv` wurden veröffentlicht, und es war einfach root zu werden. Versuchen Sie die Nutzung von SVGAlib Programmen wann immer nur möglich zu verhindern.

Kapitel 5

Absichern von Diensten, die auf Ihrem System laufen

Dienste können auf zwei Arten in einem laufenden System abgesichert werden:

- Sie so einstellen, dass auf sie nur von Zugangspunkten (Interfaces) zugegriffen werden kann, von denen es nötig ist.
- Sie so konfigurieren, dass sie nur von legitimierten Nutzern auf autorisierte Art und Weise benutzt werden können.

Dienste können durch Zugriffsbeschränkungen auf Kernel-Ebene (durch eine Firewall) eingeschränkt werden, so dass auf sie nur von bestimmten Orten aus zugegriffen werden kann. Konfigurieren Sie sie, so dass sie nur auf eine bestimmte Schnittstelle horchen (einige Dienste bieten diese Fähigkeiten vielleicht nicht). Oder verwenden Sie eine andere Methode, zum Beispiel den Linux-vserver-Patch (für 2.4.16), mit dem Prozesse auf eine bestimmte Schnittstelle gebunden werden können.

Was die Dienste angeht, die von `inetd` aufgerufen werden (`telnet`, `ftp`, `finger`, `pop3`, ...), so ist es wert zu erwähnen, dass `inetd` so konfiguriert werden kann, dass er nur auf eine bestimmte Schnittstelle reagiert (unter Verwendung der `service@ip`-Syntax). Dies ist jedoch eine nicht dokumentierte Eigenschaft. Ein Ersatz, der Meta-Daemon `xinetd`, kennt eine `bind`-Option nur für diesen Zweck. Lesen Sie dazu bitte `xinetd.conf(5)`.

```
service nntp
{
        socket_type     = stream
        protocol        = tcp
        wait            = no
        user            = news
        group           = news
        server          = /usr/bin/env
```

```
        server_args     = POSTING_OK=1 PATH=/usr/sbin/:/usr/bin:/sbin/:/bin
+/usr/sbin/snntpd logger -p news.info
        bind            = 127.0.0.1
}
```

Die folgenden Abschnitte gehen detaillierter darauf ein, wie bestimmte Dienste abhängig von der beabsichtigten Benutzung passend konfiguriert werden.

5.1 Absichern von ssh

Wenn Sie immer noch telnet statt ssh benutzen, sollten Sie dieses Handbuch kurz beiseite legen, und dies ändern. Ssh sollte anstelle von telnet für alle Fern-Logins benutzt werden. In einer Zeit, in der es leicht ist, Internet-Verkehr mitzuschnüffeln und an Klartext-Passwörter heranzukommen, sollten Sie lediglich Protokolle verwenden, die Kryptographie benutzen. Also führen Sie sofort ein `apt-get install ssh` auf Ihrem System aus.

Ermuntern Sie alle Nutzer Ihres Systems, ssh anstelle von telnet zu benutzen, oder noch besser: Deinstallieren Sie telnet/telnetd. Zusätzlich sollten Sie es vermeiden, sich mit ssh als root einzuloggen und lieber andere Methoden benutzen, um root zu werden. Wie zum Beispiel `su` oder `sudo`. Schließlich sollten Sie noch die Datei `/etc/ssh/sshd_config` für mehr Sicherheit modifizieren:

- `ListenAddress 192.168.0.1`

  Lassen Sie ssh nur auf eine bestimmte Schnittstelle hören, falls Sie mehrere haben (und ssh nicht auf allen verfügbar sein soll) oder Sie in Zukunft eine neue Netzwerkkarte einbauen werden (und keine ssh-Verbindungen auf ihr erlauben wollen).

- `PermitRootLogin no`

  Versuchen Sie so wenige Logins als Root wie möglich zu erlauben. Wenn nun jemand Root werden will, benötigt er zwei Logins. So kann das Root-Passwort nicht so leicht ausgetestet werden.

- `Port 666` oder `ListenAddress 192.168.0.1:666`

  Verändern Sie den Listen-Port, so dass ein Eindringling nicht wirklich sicher sein kann, ob ein sshd-Daemon läuft (aber beachten Sie, dass dies lediglich “Sicherheit durch Verschleierung” ist).

- `PermitEmptyPasswords no`

  Nicht gesetzte Passwörter verspotten jegliche System-Sicherheit.

- `AllowUsers alex ref ich@irgendwo`

  Erlauben Sie nur bestimmten Nutzern sich via ssh auf der Maschine einzuloggen. `user@host` kann auch verwendet werden, um einen bestimmten Benutzer user dazu zu zwingen, nur von einem bestimmten Rechner host aus zuzugreifen.

- `AllowGroups wheel admin`

  Erlauben Sie nur bestimmten Gruppenmitgliedern sich via ssh auf der Maschine einzuloggen. AllowGroups und AllowUsers haben äquivalente Verfahrensweisen, um den Zugang zu der Maschine zu verwehren. Es wird nicht überraschen, dass es sich hierbei um "DenyUsers" und "DenyGroups" handelt.

- `PasswordAuthentication yes`

  Es ist allein Ihre Wahl, was Sie hier eintragen. Es ist sicherer, Zugriff nur Nutzern zu erlauben, die ssh-Schlüssel in der `~/.ssh/authorized_keys`-Datei haben. Wenn Sie dies wollen, setzen Sie es auf "no".

- Schalten Sie jedwede Art der Authentifizierung ab, die Sie nicht wirklich benötigen, zum Beispiel `RhostsRSAAuthentication`, `HostbasedAuthentication`, `KerberosAuthentication` oder `RhostsAuthentication`. Sie sollten sie abschalten, auch wenn sie es standardmäßig bereits sind (siehe dazu die Handbuch-Seite `sshd_config(5)`).

- `Protocol 2`

  Deaktivieren Sie die Protokollversion 1, da diese einige Designschwächen hat, die es einfacher zu machen, Passwörter zu knacken. Für weitere Informationen lesen Sie a paper regarding ssh protocol problems (`http://earthops.net/ssh-timing.pdf`) oder das Xforce advisory (`http://xforce.iss.net/static/6449.php`).

- `Banner /etc/`*`eine_Datei`*

  Fügen Sie einen Bannertext (er wird aus der Datei bezogen) für Benutzer, die sich mit dem ssh-Server verbinden, hinzu. In einigen Ländern sollte das Senden einer Warnung über unautorisierten Zugriff oder Benutzerüberwachung vor dem Zugriff zu einem bestimmten System erfolgen, um sich rechtlich abzusichern.

Sie können den Zugriff auf den ssh-Server auch mittels `pam_listfile` oder `pam_wheel` in der PAM-Kontrolldatei beschränken. Zum Beispiel können Sie jeden abhalten, der nicht in der Datei `/etc/loginusers` aufgelistet ist, durch Hinzufügen folgender Zeile zu `/etc/pam.d/ssh`:

```
auth       required     pam_listfile.so sense=allow onerr=fail item=user file
```

Abschließend beachten Sie bitte, dass diese Direktiven von einer OpenSSH-Konfigurationsdatei stammen. Derzeit gibt es drei weit verbreitete ssh-Daemonen: ssh1, ssh2 und OpenSSH von den OpenBSD-Leuten. Ssh1 war der erste verfügbare ssh-Daemon und er ist noch der weit verbreitetste (Gerüchten zufolge gibt es sogar eine Windows-Version). Ssh2 hat gegenüber ssh1 viele Vorteile, abgesehen davon, dass es unter einer unfreien Lizenz veröffentlicht wurde. OpenSSH ist ein völlig freier ssh-Daemon, der sowohl ssh1 als auch ssh2 unterstützt. OpenSSH ist die Version, die installiert wird, wenn Sie auf Debian das Paket `ssh` auswählen.

Mehr Informationen, wie Sie SSH mit Unterstützung für PAM aufsetzen, finden Sie hier: security mailing list archives (`http://lists.debian.org/debian-security/2001/debian-security-200111/msg00395.html`).

5.1.1 SSH in ein Chroot-Gefängnis einsperren

Zurzeit bietet OpenSSH keine Möglichkeit, automatisch Benutzer bei der Verbindung in ein Chroot-Gefängnis einzusperren (die kommerzielle Version bietet diese Funktionalität). Wie dem auch sei, es gibt auch ein Projekt, das diese Funktionalität für OpenSSH anbietet, vergleiche `http://chrootssh.sourceforge.net`. Es ist aber aktuell noch nicht für Debian paketiert. Sie sollten stattdessen das `pam_chroot`-Modul, wie in in 'Den Nutzerzugang einschränken' auf Seite 66 beschrieben, verwenden.

In '`Chroot`-Umgebung für `SSH`' auf Seite 249 können Sie verschiedene Optionen finden, um Chroot-Umgebungen für SSH zu erstellen.

5.1.2 Ssh-Clients

Wenn Sie einen SSH-Client mit einem SSH-Server verwenden, müssen Sie sicherstellen, dass er die selben Protokolle, die vom Server erzwungen werden, unterstützt. Wenn Sie beispielsweise das Paket `mindterm` verwenden, unterstützt dies nur Protokollversion 1. Jedoch ist der sshd-Server standardmäßig so konfiguriert, nur Version 2 (aus Sicherheitsgründen) zu akzeptieren.

5.1.3 Verbieten von Dateitransfers

Wenn Sie *nicht* möchten, das Benutzer Dateien zum und vom ssh-Server übertragen, müssen Sie den Zugang zu `sftp-server` *und* zu `scp` einschränken. Sie können dies für `sftp-server` tun, indem Sie den korrekten `Subsystem` Wert in `/etc/ssh/sshd_config` eintragen. Um jedoch den Zugang zu `scp` einzuschränken, müssen Sie entweder:

- den Benutzern verbieten, sich auf dem ssh-Server einzuloggen (wie oben beschrieben entweder durch die Konfigurationsdatei oder die PAM-Konfiguration), oder
- Benutzern, denen sichere Übertragungen verwehrt sind, richtige Shells vorenthalten. Die angebotenen Shells sollten dennoch Programme sein, die Verbindungen zum ssh-Server sinnvoll gestalten, wie z.B. Menü-Programme (a la BBS). Andernfalls ist die vorherige Möglichkeit bevorzugt.

5.2 Absichern von Squid

Squid ist einer der verbreitetsten Proxy/Cache-Server und es gibt ein paar Sicherheitsaspekte, die Sie beachten sollten. Squids Standard-Konfiguration lehnt alle Anfragen von Benutzern ab. Dennoch erlaubt das Debian-Paket Zugriff von 'localhost', Sie müssen nur Ihren Browser richtig konfigurieren. Sie sollten Squid so konfigurieren, dass er Zugriffe von vertrauenswürdigen Nutzern, Computern oder Netzwerken erlaubt, indem Sie eine Zugriffs-Kontroll-Liste (ACL, Access Control List) in `/etc/squid/squid.conf` definieren. Mehr Informationen, wie Sie ACLs definieren, finden Sie im Squid User's Guide (`http://www.deckle.co.`

za/squid-users-guide/Main_Page). Ein gute deutsche Dokumentation ist das Squid-Handbuch (http://www.squid-handbuch.de/). Beachten Sie, dass Debian eine minimale Konfiguration für Squid bereitstellt, die alles verhindert, mit der Ausnahme, dass *localhost* sich mit Ihrem Proxy-Server (der standardmäßig mit dem Port 3128 läuft) verbinden kann. Sie müssen Ihre `/etc/squid/squid.conf`-Datei wie gewünscht anpassen. Die empfohlene minimale Konfiguration (mit dem Paket vertrieben) sieht wie folgt aus:

```
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl SSL_ports port 443 563
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 443 563     # https, snews
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http
acl Safe_ports port 901         # SWAT
acl purge method PURGE
acl CONNECT method CONNECT
(...)
# Erlaube nur cachemgr Zugriff von localhost
http_access allow manager localhost
http_access deny manager
# Erlaube nur purge Anfragen von localhost
http_access allow purge localhost
http_access deny purge
# Verbiete Anfragen zu unbekannten Ports
http_access deny !Safe_ports
# Verbiete CONNECT zu anderen als SSL-Ports
http_access deny CONNECT !SSL_ports
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
# And finally deny all other access to this proxy
http_access deny all
#Default:
# icp_access deny all
#
#Allow ICP queries from everyone
icp_access allow all
```

Sie sollten Squid auch entsprechend Ihren System-Ressourcen konfigurieren, inklusive Cache-Speicher (Option `cache_mem`), Lage der gecachten Dateien und der verwendeten Speichermenge auf der Platte (Option `cache_dir`).

Man beachte, dass es bei ungeeigneter Konfiguration vorkommen kann, dass jemand eine Mail über Squid weiterleitet, da die Protokolle HTTP und SMTP ein ähnliches Design haben. Squids Standardkonfiguration verweigert Zugriffe auf Port 25. Wenn Sie Verbindungen an Port 25 erlauben wollen, fügen Sie ihn einfach zu der Safe_ports-Liste hinzu. Aber dies ist *NICHT* empfohlen.

Passendes Aufsetzen und Konfigurieren des Proxy/Cache-Servers ist nur ein Teil der Absicherung Ihrer Seite. Eine andere notwendige Aufgabe ist es, Squids Log-Dateien zu analysieren, um sicher zu gehen, dass alles so arbeitet, wie es sollte. Es gibt ein paar Pakete in Debian GNU/Linux, die einem Administrator hierbei helfen können. Die folgenden Pakete sind in Debian 3.0 (Woody) und Debian 3.1 (Sarge) verfügbar:

- `calamaris` - Log-Datei-Analysator für Squid oder Oops-Proxy-Log-Dateien.
- `modlogan` - Ein modularer Log-Datei-Analysator.
- `sarg` - Squid Analysis Report Generator.
- `squidtaild` - Squid-Log-Beobachtungsprogramm.

Wenn Squid im »Accelerator Mode« betrieben wird, agiert er auch als Web-Server. Aktivieren dieser Option erhöht die Komplexität des Codes, was ihn weniger vertrauenswürdig macht. Standardmäßig ist Squid nicht dazu konfiguriert, als Web-Server zu arbeiten, Sie müssen sich darüber also keine Gedanken machen. Sie müssen sicher stellen, dass es wirklich nötig ist, wenn Sie diese Eigenschaft nutzen wollen. Weitere Informationen über den »Accelerator Mode« in Squid finden Sie im Squid User's Guide - Accelerator Mode (`http://www.deckle.co.za/squid-users-guide/Accelerator_Mode`).

5.3 Absichern von FTP

Wenn Sie wirklich FTP benutzen müssen (ohne ihn mit sslwrap zu umhüllen oder innerhalb eines SSL- oder SSH-Tunnels), sollten Sie ftp in das Home-Verzeichnis der FTP-Nutzer mit chroot einsperren, so dass sie nichts anderes sehen können als ihr eigenes Verzeichnis. Andernfalls können sie Ihr Root-Dateisystem durchlaufen, als hätten sie Shell-Zugriff darauf. Sie können die folgende Zeile in Ihre `proftpd.conf`-Datei im globalen Abschnitt hinzufügen, um die chroot-Fähigkeiten zu nutzen:

```
DefaultRoot ~
```

Starten Sie ProFTPd neu, indem Sie `/etc/init.d/proftpd restart` eingeben, und prüfen Sie, ob Sie noch aus Ihrem Home-Verzeichnis heraus kommen können.

Um ProFTPd-DoS-Angriffe durch ../../../ zu verhindern, fügen Sie die folgende Zeile Ihrer `/etc/proftpd.conf` hinzu: `DenyFilter \*.*/`

Vergessen Sie nicht, dass FTP Login- und Authentifizierungs-Passwort als Klartext sendet (dies ist kein Problem, wenn Sie einen anonymen, öffentlichen Dienst anbieten) und es gibt bessere Alternativen in Debian hierzu. Zum Beispiel `sftp` (aus dem Paket `ssh`). Es gibt auch freie Implementierungen von SSH für andere Betriebssysteme, zum Beispiel putty (http://www.chiark.greenend.org.uk/~sgtatham/putty/) oder cygwin (http://www.cygwin.com).

Wenn Sie dennoch einen FTP-Server verwalten, während Sie den Nutzern Zugriff via SSH gewähren, könnten Sie auf ein typisches Problem stoßen. Benutzer, die innerhalb eines mit SSH abgesicherten Systems auf einen anonymen FTP-Server zugreifen wollen, können versuchen, sich auf dem *FTP-Server* einzuloggen. Während der Zugriff verweigert werden wird, wird das Passwort trotzdem als Klartext über das Netz gesendet. Um dies zu verhindern, hat der ProFTPd-Entwickler TJ Saunders einen Patch erstellt, der verhindert, dass Nutzer den anonymen FTP-Server mit gültigen SSH-Zugangsdaten füttern. Mehr Informationen und den Patch finden Sie unter: ProFTPD Patches (http://www.castaglia.org/proftpd/#Patches). Dieser Patch wurde auch an Debian gesandt, vergleiche Fehler #145669 (http://bugs.debian.org/145669).

5.4 Zugriff auf das X-Window-System absichern

Heutzutage werden X-Terminals in immer mehr Firmen benutzt, wo ein Server für viele Arbeitsplätze benötigt wird. Dies kann gefährlich sein, weil Sie dem Datei-Server erlauben müssen, sich mit den X-Clients zu verbinden (X-Server aus Sicht von X. X vertauscht die Definition von Client und Server). Wenn Sie dem (sehr schlechten) Vorschlag von vielen Dokumentationen folgen, geben Sie auf Ihrer Maschine `xhost +` ein. Dies erlaubt jedem X-Client, sich mit Ihrem System zu verbinden. Für etwas bessere Sicherheit können Sie stattdessen das Kommando `xhost +Rechnername` verwenden, um den Zugriff auf bestimmte Rechner zu begrenzen.

Allerdings ist es eine viel sicherere Lösung, SSH zu benutzen, um X zu tunneln und die gesamte Sitzung zu verschlüsseln. Dies geschieht automatisch, wenn Sie sich auf eine andere Maschine via ssh einloggen. Damit dies funktioniert, müssen Sie den ssh-Client und den ssh-Server konfigurieren. Auf dem ssh-Client sollte `ForwardX11` in `/etc/ssh/ssh_config` auf `yes` gesetzt sein. Auf dem ssh-Server sollte `X11Forwarding` in `/etc/ssh/sshd_config` auf `yes` gesetzt sein und das Paket `xbase-clients` sollte installiert sein. Letzteres liegt daran, dass der SSH-Server `/usr/X11R6/bin/xauth` (bei Debian-Unstable (`/usr/bin/xauth`) verwendet, wenn er das Pseudo-X-Display aufsetzt. In den Zeiten von SSH sollten Sie die xhost-basierte Zugriffskontrolle komplett über Bord werfen.

Wenn Sie keinen X-Zugriff von anderen Maschinen benötigen, ist es für die Sicherheit am besten, die Bindung auf dem TCP-Port 6000 abzuschalten, indem Sie einfach Folgendes eingeben:

```
$ startx -- -nolisten tcp
```

Dies ist das Standard-Verhalten unter XFree 4.1.0 (der Xserver aus Debian 3.0 und 3.1). Wenn Sie XFree 3.3.6 laufen lassen (d.h. wenn Sie Debian 2.2 benutzen), können Sie `/etc/X11/xinit/xserverrc` editieren, damit Sie etwas erhalten wie:

```
#!/bin/sh
exec /usr/bin/X11/X -dpi 100 -nolisten tcp
```

Wenn Sie XDM benutzen, setzen Sie `/etc/X11/xdm/Xservers` auf `:0 local /usr/bin/X11/X vt7 -dpi 100 -nolisten tcp`. Wenn Sie GDM benutzen, stellen Sie sicher, dass die Option `DisallowTCP=true` in `/etc/gdm/gdm.conf` eingetragen ist (was standardmäßig unter Debian der Fall ist). Dies wird grundsätzlich an jede X-Befehlszeile `-nolisten tcp` anhängen [1].

Sie können außerdem die standardmäßige Zeitgrenze für die `xscreensaver`-Bildschirmsperre setzen. Auch wenn der Nutzer sie aufheben kann, sollten Sie die Konfigurationsdatei `/etc/X11/app-defaults/XScreenSaver` editieren, und die lock-Zeile von

```
*lock:                  False
```

(das ist der Standardwert unter Debian) auf

```
*lock:                  True
```

ändern.

FIXME: Add information on how to disable the screensavers which show the user desktop (which might have sensitive information).

Lesen Sie mehr zur Sicherheit von X Window in XWindow-User-HOWTO (`http://www.tldp.org/HOWTO/XWindow-User-HOWTO.html`) (`/usr/share/doc/HOWTO/en-txt/XWindow-User-HOWTO.txt.gz`).

FIXME: Add info on thread of debian-security on how to change config files of XFree 3.3.6 to do this.

5.4.1 Überprüfen Ihres Display-Managers

Wenn Sie einen Display-Manager lediglich zur lokalen Nutzung (um einen schönen graphischen Login zu haben) haben wollen, gehen Sie sicher, dass der XDMCP (X Display Manager Control Protocol) Krempel abgeschaltet ist. Unter XDM können Sie dies mit der folgenden Zeile in `/etc/X11/xdm/xdm-config` erreichen:

[1] GDM wird `-nolisten tcp` *nicht* anhängen, wenn es `-query` oder `-indirect` in der Befehlszeile findet, da sonst die Anfrage nicht funktionieren würde.

```
DisplayManager.requestPort:     0
```

Für GDM müssen Sie in Ihre gdm.conf Folgendes eintragen:

```
[xdmcp]
Enable=false
```

Normalerweise sind unter Debian alle Display-Manager so konfiguriert, dass sie standardmäßig keine XDMCP-Dienste starten.

5.5 Absichern des Druckerzugriffs (die lpd- und lprng-Problematik)

Stellen Sie sich vor, Sie kommen zur Arbeit, und der Drucker spuckt endlose Mengen von Papier aus, weil jemand eine DoS-Attacke gegen Ihren Drucker-Daemon durchführt. Unangenehm, oder?

In jeder UNIX-Druck-Architektur muss es einen Weg geben, um die Daten des Clients auf den Druck-Server zu bekommen. Traditionell machen dies `lpr` und `lp` so, dass das Client-Kommando die Daten in das Spool-Verzeichnis kopiert oder symbolisch verlinkt (weshalb diese Programme normalerweise SUID oder SGID sind).

Um jede Gefahr zu vermeiden, sollen Sie Ihren Druck-Server besonders sicher halten. Dies heißt, dass Sie Ihren Druck-Dienst so konfigurieren müssen, dass er nur Aufträge von vertrauenswürdigen Rechnern annimmt. Hierzu müssen Sie die Rechner, von denen Sie Druckaufträge entgegennehmen möchten, in die Datei `/etc/hosts.lpd` eintragen.

Allerdings akzeptiert der `lpr`-Daemon auch, wenn Sie dies getan haben, Verbindungen auf Port 515 auf jeder Schnittstelle. Sie sollten sich überlegen, ob Sie Verbindungen von Netzwerken/Rechnern, die nicht drucken dürfen, mittels Firewall abblocken wollen (der `lpr`-Daemon kann nicht so konfiguriert werden, dass er nur auf eine bestimmte IP-Adresse hört).

Sie sollten `Lprng` gegenüber `lpr` vorziehen, da er so konfiguriert werden kann, dass er Zugangskontrolle über IP beherrscht. Und Sie können spezifizieren, auf welche Schnittstelle er sich binden soll (wenn auch etwas sonderbar).

Wenn Sie Ihren Drucker nur lokal auf Ihrem System benutzen, werden Sie diesen Dienst nicht über ein Netzwerk teilen wollen. Sie sollten dann überlegen, ein anderes Druck-System, wie zum Beispiel das aus dem Paket `cups` oder PDQ (`http://pdq.sourceforge.net/`), das auf den Zugriffsrechten des Gerätes `/dev/lp0` beruht, einzusetzen.

Bei `cups` werden die Druckaufträge mit dem HTTP-Protokoll zum Server übertragen. Dadurch muss der Client nicht über spezielle Privilegien verfügen, aber dies erfordert, dass der Server auf irgendeinem Port lauscht.

Wie auch immer: Wenn Sie `cups` nur lokal benutzen möchten, können Sie ihn so konfigurieren, dass er nur auf die lokale Schleife (loopback interface) hört, indem Sie Folgendes in Ihrer `/etc/cups/cupsd.conf` ändern:

```
Listen 127.0.0.1:631
```

Es gibt noch andere Sicherheitsoptionen in dieser Konfigurationsdatei, wie zum Beispiel das Erlauben oder Verweigern von Netzwerken oder Rechnern. Wenn Sie sie allerdings nicht benötigen, belassen Sie es am besten dabei, einfach nur den Port, auf dem gehört wird, einzuschränken. `Cups` liefert auch Dokumentation über den HTTP-Port. Wenn Sie diese potenziell nützlichen Informationen einem Angreifer von außerhalb nicht enthüllen wollen (und der Port offen ist), fügen Sie außerdem Folgendes hinzu:

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

Die Konfigurationsdatei kann so angepasst werden, dass zusätzliche Fähigkeiten einschließlich SSL- und TLS-Zertifikate oder Verschlüsselung möglich werden. Die Handbücher finden Sie unter http://localhost:631/ oder `cups.org`.

FIXME: Add more content (the article on Amateur Fortress Building (`http://www.rootprompt.org`) provides some very interesting views).

FIXME: Check if PDG is available in Debian, and if so, suggest this as the preferred printing system.

FIXME: Check if Farmer/Wietse has a replacement for printer daemon and if it's available in Debian.

5.6 Absichern des Mail-Dienstes

Wenn Ihr Server kein Mail-System ist, müssen Sie wirklich keinen Mail-Daemon haben, der auf eingehende Verbindungen reagiert. Aber Sie wollen lokale Mails ausliefern, um beispielsweise Mails an den Root-User von irgendwelchen Alarmsystemen zu erhalten.

Wenn Sie `exim` haben, müssen Sie den Daemon nicht laufen lassen, um dies zu erreichen, da der Standard-`cron`-Job die Mails abarbeitet. Sehen Sie in 'Daemons abschalten' auf Seite 40 wie man dies erledigt.

5.6.1 Konfiguration eines Nullmailers

Sie mögen einen lokalen Mail-Daemon wollen, so dass er die Mails, die vom lokalen Rechner zu einem anderen System geschickt wurden, versenden kann. Dies ist üblich, wenn Sie eine Anzahl von Systemen zu administrieren haben und nicht zu jedem von diesen eine Verbindung aufbauen wollen, um die dort lokal verschickten Mails zu lesen. Genau wie all das Protokollieren eines jeden individuellen Systems durch einen zentralen syslog-Server zentralisiert werden kann, so kann auch Mail zu einem zentralen Mail-Server gesandt werden.

Solch ein *nur sendendes* System sollte sorgfältig dafür eingerichtet werden. Der Daemon kann ebenso konfiguriert werden, nur an der Loopback-Adresse zu lauschen.

Die folgenden Konfigurationsschritte müssen nur zur Konfiguration des `exim`-Pakets in der Debian 3.0 Version vorgenommen werden. Wenn Sie eine neuere Version verwenden (wie z.B. 3.1, das `exim4` verwendet), so wurde das Installationssystem verbessert, so dass, wenn der Mail-Transport-Agent konfiguriert wurde nur lokale Mails zu versenden, es automatisch nur Verbindungen vom lokalen Rechner und keine entfernten Verbindungen zulässt.

In einem Debian 3.0 System mit `exim` muss man den SMTP-Daemon aus `inetd` wie folgt entfernen:

```
$ update-inetd --disable smtp
```

und den Mail-Daemon so konfigurieren, dass er nur auf die lokale Schleife achtet. In `exim` (dem Standard-Mail-Transport-Agent (MTA) unter Debian) tun Sie dies, indem Sie in der Datei `/etc/exim.conf` die Zeile

```
local_interfaces = "127.0.0.1"
```

hinzufügen.

Starten Sie beide Daemonen neu (inetd und exim) und exim wird lediglich auf den Socket 127.0.0.1:25 lauschen. Seien Sie vorsichtig und deaktivieren Sie erst inetd, oder exim wird nicht neu starten, da der inetd-Daemon bereits eingehende Verbindungen behandelt.

Bei `postfix` editieren Sie `/etc/postfix/main.conf`:

```
inet_interfaces = localhost
```

Wenn Sie lediglich lokale Mails wollen, ist dieses Herangehen besser als den Mailer-Daemon in einen tcp-Wrapper zu hüllen oder Firewall-Regeln einzufügen, die den Zugang für alle limitieren. Wenn Sie jedoch auch auf andere Schnittstellen reagieren müssen, sollten Sie überlegen, ihn vom inetd aufrufen zu lassen und einen tcp-Wrapper einzusetzen, so dass eingehende Verbindungen gegen `/etc/hosts.allow` und `/etc/hosts.deny` geprüft werden. Außerdem werden Sie vor unautorisierten Zugriffsversuchen gegen Ihren Mail-Daemon durch angemessenes Protokollieren gewarnt werden wollen.

In jedem Fall können Sie Mail-Zustellversuche auf dem SMTP-Level ablehnen, indem Sie die `/etc/exim/exim.conf` abändern, damit Sie Folgendes enthält:

```
receiver_verify = true
```

Auch wenn Ihr Mail-Server keine Mails zustellen wird, ist diese Konfiguration für den Relay-Tester auf `http://www.abuse.net/relay.html` nötig, um festzustellen, dass Ihr Server *nicht* relaisfähig ist.

Wenn Sie Mails nur weiterleiten möchten, können Sie in Erwägung ziehen, den Mail-Daemon durch Programme zu ersetzen, die *nur* zum Weiterleiten der Mail zu einem entfernten Mail-Server konfiguriert werden können. Debian stellt zurzeit `ssmtp` und `nullmailer` für diese Zwecke zur Verfügung. Auf jeden Fall können Sie für sich selbst alle von Debian angebotenen Mail-Transport-Agents testen [2] und sehen, welcher davon am besten auf Ihr System zugeschnitten ist.

5.6.2 Anbieten von sicherem Zugang zu Mailboxen

Wenn Sie entfernten Zugriff auf Mailboxen erlauben wollen, gibt es eine Anzahl von möglichen POP3- und IMAP-Daemonen.[3] Wenn Sie IMAP-Zugriff anbieten, beachten Sie jedoch, dass es ein allgemeines Dateizugriffsprotokoll ist. Es kann das Äquivalent zu einem Shell-Zugang werden, da Benutzer in der Lage sein könnten, Zugang zu beliebigen Dateien zu erhalten, auf die sie durch ihn zugreifen können.

Versuchen Sie beispielsweise, `{server.com}/etc/passwd` als Ihren Eingabepfad zu konfigurieren. Wenn dies gelingt, ist Ihr IMAP-Daemon nicht richtig konfiguriert, um diese Art von Zugriff zu verhindern.

Unter den IMAP-Servern in Debian vermeidet der `cyrus`-Server (im Paket `cyrus-imapd`) dies, indem er den gesamten Zugriff zu einer Datenbasis in einem beschränkten Teil des Dateisystems limitiert. Auch `uw-imapd` (installieren Sie entweder das `uw-imapd`- oder besser, wenn Ihre IMAP-Clients es unterstützen, das `uw-imapd-ssl`-Paket) kann konfiguriert werden, das Mailverzeichnis der Benutzer in ein Chroot-Gefängnis einzusperren, dies ist jedoch nicht standardmäßig aktiviert. Die angebotene Dokumentation enthält mehr Informationen, wie man dies konfiguriert.

Es ist ebenso empfehlenswert, einen IMAP-Server laufen zu haben, der keine neuen Benutzer im lokalen System erfordert (dies würde auch einen Shell-Zugang ermöglichen). Sowohl `courier-imap` (für IMAP) und `courier-pop`, `teapop` (für POP3) und `cyrus-imapd` (für POP3 und IMAP) bieten Server mit Authentifizierungsmethoden neben den lokalen Benutzerkonten. `cyrus` kann alle Authentifizierungsmethoden, die mittels PAM konfiguriert werden können, verwenden, währenddessen `teapop` Datenbanken (wie `postgresql` und `mysql`) für die Benutzerauthentifizierung nutzen kann.

FIXME: Check: `uw-imapd` might be configured with user authentication through PAM too.

[2] Die Liste der in Debian verfügbaren Mail-Daemons erhalten Sie wie folgt:

```
$ apt-cache search mail-transport-agent
```

Die Liste wird `qmail` nicht enthalten, da dies nur im Quellcode im Paket `qmail-src` vertrieben wird.

[3] Eine Liste von Servern/Daemonen die diese Protokolle in Debian anbieten, kann wie folgt erhalten werden:

```
$ apt-cache search pop3-server $ apt-cache search imap-server
```

5.6.3 Sicherer Empfang von Mails

Das Lesen und Empfangen von Mails ist das gebräuchlichste Klartext-Protokoll. Wenn Sie POP3 oder IMAP benutzen, um Ihre Mails zu erhalten, senden Sie ein Klartext-Passwort über das gesamte Netz, so dass ziemlich jeder Ihre Mails von nun an lesen kann. Benutzen Sie stattdessen SSL (Secure Sockets Layer), um Ihre Mails zu empfangen. Wenn Sie einen Shell-Account auf dem Rechner, der als POP oder IMAP-Server agiert, haben, ist die andere Alternative SSH. Hier ist eine beispielhafte `fetchmailrc` um dies zu zeigen:

```
poll mein-imap-mailserver.org via "localhost"
  with proto IMAP port 1236
      user "ref" there with password "hackmich" is alex here warnings 3600
    folders
      .Mail/debian
    preconnect 'ssh -f -P -C -L 1236:my-imap-mailserver.org:143 -l ref
     mein-imap-mailserver.org sleep 15 </dev/null > /dev/null'
```

Die wichtige Zeile ist die preconnect-Zeile. Sie startet eine SSH-Verbindung und erstellt den notwendigen Tunnel, durch den automatisch alle Verbindungen zum lokalen Port 1236 verschlüsselt an den IMAP-Mail-Server weitergeleitet werden. Eine andere Möglichkeit wäre es, `fetchmail` mit SSL-Unterstützung zu benutzen.

Wenn Sie verschlüsselte Mail-Dienste wie POP oder IMAP anbieten möchten, `apt-get install stunnel` und starten Sie Ihren Daemon auf diese Weise:

```
stunnel -p /etc/ssl/certs/stunnel.pem -d pop3s -l /usr/sbin/popd
```

Dieses Kommando umhüllt den angegebenen Daemon (-l) an den Port (-d) und nutzt ein bestimmtes SSL-Zertifikat (-p).

5.7 Sichern von BIND

Es gibt verschiedene Dinge, mit denen Sie sich auseinander setzen sollten, um einen Domain-Server-Daemon abzusichern, die ähnlich zu den Überlegungen sind, wie man einen anderen Dienst absichert:

- Konfigurieren Sie den Daemon selbst, so dass er von außen nicht missbraucht werden kann (siehe auch 'Bind-Konfiguration um Missbrauch zu verhindern' auf der nächsten Seite). Dies schließt das Einschränken von Abfragen durch Clients ein: Zonen-Transfers und rekursive Abfragen.
- Einschränken des Zugriffs des Daemon auf den Server selbst, so dass dem Schaden für das System im Falle eines Einbruchs Grenzen gesetzt sind. Hierzu gehört auch, den Daemon als nicht-privilegierten Benutzer laufen zu lassen (siehe 'Ändern des BIND-Benutzers' auf Seite 112) und ihn in ein Chroot-Gefängnis einzusperren (siehe 'Chroot-Gefängnis für Name-Server' auf Seite 114).

5.7.1 Bind-Konfiguration um Missbrauch zu verhindern

Sie sollten einige Informationen, die von außen über den DNS-Server abgefragt werden können, zurückhalten, so dass man nicht wertvolle Informationen über Ihre Organisation, die Sie nicht herausgeben wollen, abfragen kann. Dies schließt die folgenden Optionen mit ein: *allow-transfer*, *allow-query*, *allow-recursion* und *version*. Sie können dies in dem globalen Abschnitt tun (so wird es auf alle Zonen angewandt) oder jeweils pro Zone. Dies ist im Paket `bind-doc` dokumentiert. Sobald das Paket installiert ist, können Sie hierzu mehr in `/usr/share/doc/bind/html/index.html` lesen.

Stellen Sie sich vor, Ihr Server ist mit dem Internet und Ihrem internen Netzwerk (Ihre interne IP ist 192.168.1.2) verbunden – ein einfacher Server im heimischen Netzwerk. Sie möchten keinen Dienst im Internet anbieten und lediglich DNS-Abfragen von Ihren internen Rechnern erlauben. Sie können dies einschränken, indem Sie Folgendes in Ihre `/etc/bind/named.conf` aufnehmen:

```
options {
            allow-query { 192.168.1/24; } ;
            allow-transfer { none; } ;
            allow-recursion { 192.168.1/24; } ;
            listen-on { 192.168.1.2; } ;
            forward { only; } ;
            forwarders { A.B.C.D; } ;
};
```

Die Option *listen-on* bewirkt, dass sich DNS nur auf die Schnittstelle bindet, die die interne Adresse hat. Aber sogar wenn diese Schnittstelle Verbindung zum Internet hat (zum Beispiel weil Sie NAT benutzen), werden Abfragen nur akzeptiert, wenn sie von internen Hosts kommen. Wenn das System mehrere Schnittstellen hat und Sie kein *listen-on* gesetzt haben, könnten zwar nur interne Nutzer Abfragen starten, aber, da der Port für Angreifer von außen ansprechbar ist, könnten sie versuchen, den DNS zum Absturz zu bringen (oder durch Speicher-Überlauf-Attacken auszunutzen). Sie könnten ihn sogar dazu bringen, lediglich auf 127.0.0.1 zu hören, wenn Sie den DNS-Service nicht für ein anderes System anbieten wollen.

Der version.bind-Eintrag in der chaos class enthält die Version des derzeit laufenden Bind-Prozesses. Diese Information wird oft von automatischen Scannern und bösartigen Individuen dazu verwendet herauszufinden, ob ein `bind` für eine bestimmte Attacke verwundbar ist. Indem Sie falsche oder gar keine Informationen im version.bind-Eintrag zur Verfügung stellen, minimieren Sie die Wahrscheinlichkeit, dass jemand Ihren Server aufgrund der publizierten Version attackieren wird. Um Ihre eigene Version anzugeben, benutzen Sie die *Version*-Direktive in der folgenden Art:

```
 options { ... verschiedene andere Optionen ...
version "Nicht verfuegbar."; };
```

Das Ändern des version.bind-Eintrags schützt eigentlich nicht gegen Attacken, aber Sie können es als sinnvolle Schutzvorrichtung ansehen.

Eine beispielhafte `named.conf`-Konfigurationsdatei könnte so aussehen:

```
acl internal {
        127.0.0.1/32;           // localhost
        10.0.0.0/8;             // intern
        aa.bb.cc.dd;            // eth0 IP
};

acl friendly {
        ee.ff.gg.hh;            // slave DNS
        aa.bb.cc.dd;            // eth0 IP
        127.0.0.1/32;           // localhost
        10.0.0.0/8;             // intern
};

options {
        directory "/var/cache/bind";
        allow-query { internal; };
        allow-recursion { internal; };
        allow-transfer { none; };
};
// Ab hier bis zur meineseite.bogus Zone
// ist alles im Grunde die unveränderte Debian-Standardeinstellung
logging {
        category lame-servers { null; };
        category cname { null; };
};

zone "." {
        type hint;
        file "/etc/bind/db.root";
};

zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
```

```
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};

// Zone, die ich selbst hinzugefügt habe
zone "meineseite.bogus" {
        type master;
        file "/etc/bind/named.meineseite";
        allow-query { any; };
        allow-transfer { friendly; };
};
```

Bitte prüfen Sie (erneut) die Debian-Fehler-Datenbank (BTS) bezüglich Bind, insbesondere Bug #94760 (regarding ACLs on zone transfers) (`http://bugs.debian.org/94760`). Fühlen Sie sich ruhig dazu ermutigt, zu diesem Bugreport beizutragen, wenn Sie glauben, nützliche Informationen hinzufügen zu können.

5.7.2 Ändern des BIND-Benutzers

Bezüglich der Beschränkung von BINDs Privilegien müssen Sie beachten, dass, wenn Sie BIND als nicht-root Benutzer laufen lassen, BIND neue Netzwerk-Schnittstellen nicht automatisch entdecken kann, zum Beispiel wenn Sie eine PCMCIA-Karte in Ihr Notebook stecken. Lesen Sie die Datei `README.Debian` in Ihrer named-Dokumentation (`/usr/share/doc/bind/README.Debian`) für mehr Informationen hierzu. Es gab in letzter Zeit viele Sicherheitsprobleme mit BIND, so dass es nützlich ist, den Benutzer zu wechseln, wenn es möglich ist. Wir werden die Schritte, die dazu nötig sind, detailliert aufführen. Wenn Sie dies automatisch machen lassen wollen, können Sie das Skript in 'Beispielskript, um die Standard-Installation von Bind zu ändern' auf Seite 241 ausprobieren.

Um BIND unter einem anderen Benutzer laufen zu lassen, müssen Sie zunächst einen separaten Benutzer und eine separate Gruppe dafür erstellen (es ist *keine* gute Idee, für alle Dienste, die Sie nicht als root laufen lassen, den Benutzer nobody und die Gruppe nogroup zu benutzen). In diesem Beispiel wird der Benutzer und die Gruppe `named` benutzt. Sie können diese anlegen, indem Sie die folgenden Kommandos eingeben:

```
addgroup named
adduser --system --home /home/named --no-create-home --ingroup named \
      --disabled-password --disabled-login named
```

Beachten Sie, dass der Benutzer `named` sehr eingeschränkt ist. Wenn Sie – aus welchen Gründen auch immer – ein weniger eingeschränktes Setup haben möchten, benutzen Sie:

```
adduser --system --ingroup named named
```

Editieren Sie nun `/etc/init.d/bind` mit Ihrem Lieblingseditor und ändern Sie die Zeile, die mit

```
start-stop-daemon --start
```

anfängt zu[4]:

```
start-stop-daemon --start --quiet --exec /usr/sbin/named -- -g named -u named
```

Ändern Sie die Rechte der Dateien, die von Bind verwendet werden, inklusive `/etc/bind /rndc.key`:

```
-rw-r-----    1 root     named          77 Jan  4 01:02 rndc.key
```

und wo bind seine PID-Datei erzeugt, z.B. indem Sie `/var/run/named` anstatt von `/var/run` verwenden:

```
$ mkdir /var/run/named
$ chown named.named /var/run/named
$ vi /etc/named.conf
[ ... ändern Sie die Konfigurationsdatei, um diesen neuen Pfad zu verwenden .
options { ...
        pid-file "/var/run/named/named.pid";
};
[ ... ]
```

Außerdem müssen Sie, um zu verhindern, dass irgendetwas als root läuft, die `reload`-Zeile auskommentieren:

```
reload)
       /usr/sbin/ndc reload
```

und in Folgendes ändern:

```
reload)
        $0 stop
        sleep 1
        $0 start
```

[4] Beachten Sie, dass Sie abhängig von Ihrer Bind-Version die Option `-g` nicht haben, höchstwahrscheinlich wenn Sie bind9 von Woody (9.2.1-2.woody) oder Sarge (9.2.4) installiert haben.

Beachten Sie: Abhängig von Ihrer Debian-Version, müssen Sie vielleicht auch die `restart`-Zeile ändern. Dies wurde in der Version `1:8.3.1-2` von Debians BIND-Paket repariert.

Alles, was Sie jetzt noch tun müssen, ist, bind mittels `/etc/init.d/bind restart` neu zu starten und dann Ihr Syslog auf zwei Einträge wie die folgenden zu prüfen:

```
Sep  4 15:11:08 nexus named[13439]: group = named
Sep  4 15:11:08 nexus named[13439]: user = named
```

Voilà! Ihr named läuft *nicht mehr* als root. Wenn Sie mehr Informationen darüber lesen wollen, warum BIND nicht als nicht-root Benutzer auf Debian-Systemen läuft, sehen Sie bitte in der Fehlerdatenbank zu Bind nach, insbesondere Bug #50013: bind should not run as root (`http://bugs.debian.org/50013`) und Bug #132582: Default install is potentially insecure (`http://bugs.debian.org/132582`), Bug #53550 (`http://bugs.debian.org/53550`), Bug #52745 (`http://bugs.debian.org/52745`) und Bug #128129 (`http://bugs.debian.org/128129`). Fühlen Sie sich ruhig dazu ermuntert, etwas zu den Fehlerbeschreibungen beizutragen, wenn Sie denken, nützliche Informationen hinzufügen zu können.

5.7.3 Chroot-Gefängnis für Name-Server

Um die größtmögliche BIND-Sicherheit zu erreichen, müssen Sie nun ein Chroot-Gefängnis (siehe 'Allgemeine chroot- und suid-Paranoia' auf Seite 118) um Ihren Daemon herum bauen. Es gibt einen einfachen Weg, dies zu erreichen: Die Option `-t` (siehe die Handbuchseite `named(8)` oder Seite 100 von Bind's 9 Dokumentation (PDF) (`http://www.nominum.com/content/documents/bind9arm.pdf`)). Dies wird Bind selbst in ein bestimmtes Verzeichnis chrooten lassen, ohne dass Sie ein eigenes Chroot-Gefängnis aufsetzen und sich Sorgen um dynamische Bibliotheken machen müssen. Die einzigen Dateien, die in diesem Chroot-Gefängnis benötigt werden, sind:

```
dev/null
etc/bind/       - sollte die named.conf und alle Server-Zonen enthalten
sbin/named-xfer - wenn Sie Namen transferieren
var/run/named/  - sollte die PID und den Cache des Name-Servers (falls es
                  ihn gibt) enthalten. Dieses Verzeichnis muss für
                  den named-User schreibbar sein.
var/log/named   - Wenn Sie in eine Datei protokollieren, muss dies
                  für den named-User schreibbar sein.
dev/log         - syslogd sollte hierauf hören, wenn named so
                  konfiguriert ist, dass er darüber protokolliert.
```

Damit Ihr Bind-Daemon vernünftig läuft, braucht er bestimmte Zugriffsrechte auf die named-Dateien. Dies ist eine einfache Angelegenheit, da die Konfigurationsdateien immer in `/etc/named/` liegen. Beachten Sie, dass er lediglich Lesezugriff benötigt, es sei denn, es handelt sich um einen sekundären oder zwischenspeichernden (Cache) Name-Server. Wenn dies

der Fall ist, müssen Sie ihm Lese- und Schreibzugriff auf die notwendigen Zonen gewähren (damit Zonen-Transfers vom primären Server funktionieren).

Mehr Informationen über das Chrooten von Bind finden Sie unter Chroot-BIND-HOWTO (http://www.tldp.org/HOWTO/Chroot-BIND-HOWTO.html) (betrifft Bind 9) und Chroot-BIND8-HOWTO (http://www.tldp.org/HOWTO/Chroot-BIND8-HOWTO.html) (betrifft Bind 8). Diese Dokumente sollten auch nach der Installation des Paketes `doc-linux-text` (Text-Version) oder `doc-linux-html` (HTML-Version) verfügbar sein. Ein anderes nützliches Dokument ist http://web.archive.org/web/20011024064030/http://www.psionic.com/papers/dns/dns-linux.

Wenn Sie für Bind ein komplettes Chroot-Gefängnis aufsetzen (d.h. Sie benutzen nicht nur `-t`), stellen Sie sicher, dass Sie die folgenden Dateien darin haben:[5]

```
dev/log -  syslogd sollte hierauf hören
dev/null
etc/bind/named.conf
etc/localtime
etc/group - mit einer einzigen Zeile: "named:x:GID:"
etc/ld.so.cache - mit ldconfig erstellt
lib/ld-2.1.3.so
lib/libc-2.1.3.so
lib/ld-linux.so.2 - symbolischer Link auf ld-2.1.3.so
lib/libc.so.6 - symbolischer Link auf libc-2.1.3.so
sbin/ldconfig - kann gelöscht werden, nachdem Chroot aufgesetzt wurde
sbin/named-xfer - wenn Sie Namen transferieren
var/run/
```

Sorgen Sie auch dafür, dass `syslogd` auf `$CHROOT/dev/log` achtet, so dass der Name-Server seine syslog-Einträge in das lokale System-Protokoll schreiben lassen kann.

Wenn Sie Probleme mit dynamischen Bibliotheken vermeiden wollen, können Sie Bind statisch kompilieren. Sie können hierzu `apt-get` mit der `source` Option benutzen. Es kann sogar die Pakete herunterladen, die Sie zum Kompilieren benötigen. Sie müssten etwas ähnliches wie das Folgende tun:

```
$ apt-get source bind
# apt-get build-dep bind
$ cd bind-8.2.5-2
(editieren Sie src/port/linux/Makefile, so dass CFLAGS die Option
'-static' beinhaltet)
$ dpkg-buildpackage -rfakeroot -uc -us
$ cd ..
# dpkg -i bind-8.2.5-2*deb
```

[5] Diese Einstellungen wurden für Bind 8.4.3 von Debian 2.2 (Potato) getestet.

Nach der Installation werden Sie die Dateien in das Chroot-Gefängnis verschieben müssen.[6] Sie können die `init.d`-Skripte in `/etc/init.d` lassen, so dass das System automatisch den Name-Server starten wird, aber editieren Sie sie, indem Sie bei den `start-stop-daemon` Aufrufen in diesen Skripten `--chroot /location_of_chroot` hinzufügen.

Für weitere Informationen wie man Chroot-Gefängnisse aufsetzt, siehe 'Allgemeine chroot- und suid-Paranoia' auf Seite 118.

FIXME: Füge Informationen aus folgenden Quellen ein: http://people.debian.org/~pzn/howto/chroot-bind.sh.txt, http://www.cryptio.net/~ferlatte/config/ (Debian-spezifisch), http://web.archive.org/web/20021216104548/http://www.psionic.com/papers/whitep01.html und http://csrc.nist.gov/fasp/FASPDocs/NISTSecuringDNS.htm.

5.8 Absichern von Apache

FIXME: Add content: modules provided with the normal Apache installation (under /usr/lib/apache/X.X/mod_*) and modules that can be installed separately in libapache-mod-XXX packages.

Sie können den Zugriff auf Ihren Apache-Server einschränken, wenn Sie ihn nur intern benutzen wollen (zum Beispiel zu Testzwecken, oder um auf die `doc-central`-Archive zuzugreifen, etc.) und nicht wollen, dass von außen auf ihn zugegriffen werden kann. Um dies zu tun, benutzen Sie die `Listen` oder `BindAddress` Direktiven in der Datei `/etc/apache/http.conf`.

Benutzen von Listen:

```
Listen 127.0.0.1:80
```

Benutzen von BindAddress:

```
BindAddress 127.0.0.1
```

Starten Sie anschließend Apache mit `/etc/init.d/apache restart` neu, und Sie werden sehen, dass er nur auf die lokale Schleife achtet.

In jedem Fall sollten Sie, wenn Sie nicht die ganze Funktionalität, die Apache zur Verfügung stellt, benutzen wollen, mal einen Blick auf die anderen Web-Server aus Debian werfen, zum Beispiel `dhttpd`.

Die Apache Documentation (http://httpd.apache.org/docs/misc/security_tips.html) stellt viele Informationen zu Sicherheitsmaßnahmen, die Sie auf einem Apache

[6] Es sei denn, Sie benutzen die `instdir`-Option, wenn Sie `dpkg` aufrufen, aber dann könnte das chroot-Gefängnis etwas komplizierter werden.

Web-Server anwenden können, bereit (die gleichen Informationen erhalten Sie unter Debian auch durch das Paket `apache-doc`).

Mehr Informationen zu weiteren Restriktionen von Apache durch Einrichten chroot-Gefängnisses wird in '`Chroot`-Umgebung für `Apache`' auf Seite 267 bereitgestellt.

5.8.1 Verhindern, dass Benutzer Web-Inhalte veröffentlichen

Die Standard-Apache-Installation in Debian erlaubt Benutzern Inhalt unter `$HOME/public_html` bereitzustellen. Dieser Inhalt kann aus aus der Ferne mit einer URL wie http://Ihr_Apache_Server/~benutzer abgegriffen werden.

Wenn Sie dies nicht erlauben wollen, müssen Sie in der Konfigurationsdatei `/etc/apache/http.conf` (von Apache 1.3) folgendes Module auskommentieren:

```
LoadModule userdir_module /usr/lib/apache/1.3/mod_userdir.so
```

Wenn Sie Apache 2.0 verwenden, müssen Sie die Datei `/etc/apache2/mods-enabled/userdir.load` entfernen oder die Standardkonfiguration einschränken, indem Sie `/etc/apache2/mods-enabled/userdir.conf` bearbeiten.

Falls allerdings das Modul statisch verlinkt wurde (Sie können die Module, die einkompiliert wurden, mittels `apache -l` überprüfen), müssen Sie das Folgende der Konfigurationsdatei von Apache hinzufügen:

```
Userdir disabled
```

Beachten Sie, dass `disabled` nur ab Version 1.3 von Apache verfügbar ist [7].

Ein Angreifer kann immer noch die Benutzer herausfinden, da die Antwort des Web-Servers *403 Permission Denied* und nicht *404 Not available* lautet. Mit dem Rewrite-Modul können Sie das verhindern.

5.8.2 Rechte von Log-Datei

Apache-Log-Dateien gehören seit 1.3.22-1 dem Benutzer 'root' und der Gruppe 'adm' mit den Rechtebits 640. Diese Rechte ändern sich nach einer Rotation. Ein Eindringling, der das System über den Web-Server erreicht hat, kann so alte Log-Datei-Einträge nicht (ohne Rechteerweiterung) entfernen.

[7] Falls Sie eine ältere Version von Apache einsetzen (wovon abgeraten wird, da sie seit der Veröffentlichung von Apache 1.3 in Woody nicht mehr unterstützt wird), müssen Sie der Konfigurationsdatei Folgendes hinzufügen:

```
<Directory /home/*/public_html> AllowOverride None Order deny,allow Deny from all
</Directory>
```

5.8.3 Veröffentlichte Web-Dateien

Apache-Dateien befinden sich unterhalb von `/var/www`. Direkt nach der Installation bietet die Standardseite einige Informationen zu dem System (hauptsächlich dass es ein Debian-System ist, auf welchem Apache läuft). Die Standard-Webseiten gehören standardmäßig dem Benutzer root und der Gruppe root, währenddessen der Apache-Prozess als Benutzer www-data und Gruppe www-data läuft. Dies sollte es Angreifern, die in das System durch den Web-Server eindringen, schwerer machen, die Site zu verunstalten. Sie sollten natürlich die Standard-Webseiten (die Informationen, die Sie der Außenwelt vorenthalten wollen, enthalten können) durch Ihre eigenen ersetzen.

5.9 Absichern von Finger

Wenn Sie den Finger-Dienst laufen lassen wollen, fragen Sie sich bitte zuerst, ob Sie das auch wirklich tun müssen. Wenn Sie es müssen, werden Sie feststellen, dass Debian viele Finger-Daemonen zur Verfügung stellt (hier die Ausgabe von `apt-cache search fingerd`):

- cfingerd - Konfigurierbarer finger-Daemon
- efingerd - Ein weiterer Unix-finger-Daemon mit anpassbarer Ausgabe
- ffingerd - Ein sicherer finger-Daemon
- fingerd - Remote-User Informationsserver
- xfingerd - BSD-ähnlicher finger-Daemon mit qmail-Unterstützung

`ffingerd` ist der empfohlene finger-Daemon, wenn Sie vorhaben, einen öffentlichen Dienst anzubieten. In jedem Fall sind Sie dazu angespornt, wenn Sie ihn über inetd, xinetd oder tcpserver laufend aufzusetzen: Schränken Sie die Anzahl der Prozesse die gleichzeitig laufen dürfen ein. Schränken Sie den Zugriff auf den Finger-Daemon von bestimmten Hosts ein (indem Sie tcp-wrapper benutzen) und lassen Sie ihn nur auf die Schnittstellen lauschen, auf die er achten muss.

5.10 Allgemeine chroot- und suid-Paranoia

`chroot` ist eine der mächtigsten Möglichkeiten, einen Daemon oder einen Benutzer oder einen anderen Dienst zu beschränken. Denken Sie einfach an ein Gefängnis um Ihr Ziel, das das Ziel nicht verlassen kann (normalerweise, es gibt aber einige Bedingungen, die einem einen Ausbruch aus solch einem Gefängnis gestatten). Wenn Sie einem Benutzer oder einem Dienst nicht trauen, können Sie eine modifizierte root-Umgebung für ihn erzeugen. Dies kann einiges an Plattenplatz benötigen, da Sie alle benötigten Programme ebenso wie Bibliotheken in das Gefängnis kopieren müssen. Aber danach ist die Wirkung des Schadens, selbst wenn der Benutzer etwas bösartiges macht, auf das Gefängnis beschränkt.

Viele Dienste, die als Daemonen laufen, können von dieser Vorgehensweise profitieren. Die Daemonen, die Sie mit Ihrer Debian-Distribution installieren, laufen jedoch nicht standardmäßig in einem chroot-Gefängnis.[8]

Dies beinhaltet: Name-Server (wie `bind`), Web-Server (wie `apache`), Mail-Server (wie `sendmail`) und FTP-Server (wie `wu-ftpd`). Wahrscheinlich ist es nur fair zu sagen, dass die Komplexität von BIND der Grund dafür ist, warum er in den letzten Jahren so oft für Attacken verwundbar war (vergleiche 'Sichern von BIND' auf Seite 109).

Jedoch bietet Debian einige Software an, die helfen kann, `chroot`-Umgebungen aufzubauen. Sehen Sie 'Automatisches Erstellen von Chroot-Umgebungen' auf dieser Seite.

Wenn Sie irgendwelche Dienste in Ihrem System laufen lassen, sollten Sie dies so sicher wie nur möglich tun. Dies beinhaltet: Entziehen von root-Privilegien, Starten in beschränkten Umgebungen (wie ein chroot-Gefängnis) oder Ersetzen durch ein sichereres Äquivalent.

Seien Sie jedoch gewarnt, dass aus einem `chroot`-Gefängnis ausgebrochen werden kann, wenn der Benutzer, der im Inneren läuft, der Superuser ist. Sie müssen also sicherstellen, dass der Dienst als nicht privilegierter Benutzer läuft. Durch Einschränken seiner Umgebung schränken Sie die Welt-lesbaren/ausführbaren Dateien, auf die der Dienst zugreifen kann, ein. Auf diese Weise limitieren Sie die Möglichkeiten einer Rechteerweiterung durch lokale Sicherheitsverwundbarkeiten des Systems. Selbst in dieser Situation können Sie nicht völlig sicher sein, dass es für einen cleveren Angreifer keinen Weg gibt, irgendwie aus dem Gefängnis auszubrechen. Verwenden von nur sicheren Server-Programmen, die einen guten Ruf bezüglich Sicherheit haben, ist eine zusätzliche gute Sicherheitsmaßnahme. Selbst kleinste Löcher wie offene Datei-Handle können von einem versierten Angreifer zum Einbruch in das System verwendet werden. Schließlich war `chroot` nicht als Sicherheitstool gedacht, sondern als ein Testwerkzeug.

5.10.1 Automatisches Erstellen von Chroot-Umgebungen

Es gibt verschiedene Programme, um Server und Dienste automatisch in ein Chroot-Gefängnis einzusperren. Debian bietet zurzeit (akzeptiert im Mai 2002) Wietse Venemas `chrootuid` im Paket `chrootuid`, ebenso wie `compartment` und `makejail` an. Diese Programme können verwendet werden, um eine eingeschränkte Umgebung zum Ausführen beliebiger Programme aufzusetzen (`chrootuid` erlaubt es Ihnen sogar, es unter einem eingeschränkten Benutzer laufen zu lassen).

Einige dieser Werkzeuge können verwendet werden, um das Chroot-Gefängnis leicht aufzusetzen. Zum Beispiel kann das `makejail`-Programm ein chroot-Gefängnis mit kurzen Konfigurationsdateien erzeugen und aktualisieren. (Es bietet Beispielskonfigurationsdateien für `bind`, `apache`, `postgresql` und `mysql`.) Es versucht alle Dateien, die vom Daemon benötigt werden, mittels `strace`, `stat` und Debians Paketabhängigkeiten zu bestimmen und in das Gefängnis zu installieren. Weitere Information gibt es unter `http://www.floc.net/makejail/`. `Jailer` ist ein ähnliches Werkzeug und kann von `http://www.balabit.hu/downloads/jailer/` heruntergeladen werden und ist auch als Debian-Paket verfügbar.

[8] Es wird versucht, diese mit *minimalen Rechten* laufen zu lassen, was beinhaltet, Daemonen unter ihren eigenen Benutzern, anstatt unter root, laufen zu lassen.

5.11 Allgemeine Klartextpasswort-Paranoia

Sie sollten versuchen, jeden Netzwerk-Dienst, der seine Passworte als Klartext über das Netz sendet oder empfängt, wie zum Beispiel FTP/Telnet/NIS/RPC, vermeiden. Der Autor empfiehlt jedem, ssh anstelle von telnet und ftp zu verwenden.

Vergessen Sie jedoch nicht, dass die Migration von telnet zu ssh die Sicherheit in keinster Weise erhöht, wenn Sie weiterhin Klartext-Protokolle verwenden. Am besten wäre es ftp, telnet, pop, imap und http zu entfernen und durch ihre entsprechenden verschlüsselten Dienste zu ersetzen. Sie sollten in Erwägung ziehen von diesen Diensten zu deren SSL-Versionen zu wechseln: ftp-ssl, telnet-ssl, pop-ssl, https, ...

Die meisten der oben aufgelisteten Tipps gelten für jedes Unix-System (Sie werden sie in jedem anderen sicherheitsrelevanten Dokument, das Sie jemals lesen, wiederfinden, wenn es sich auf Linux und andere Unices bezieht).

5.12 NIS deaktivieren

Sie sollten, wenn möglich, nicht NIS, den Network Information Service, benutzen, da er das gemeinsame Nutzen von Passwörtern erlaubt. Dies kann sehr unsicher sein, wenn Ihr Setup fehlerhaft ist.

Wenn Sie Passwörter zwischen verschiedenen Maschinen teilen müssen, sollten Sie andere Alternativen in Erwägung ziehen. Zum Beispiel können Sie einen LDAP-Server aufsetzen und PAM auf Ihrem System so konfigurieren, dass es den LDAP-Server zur Benutzer-Authentifizierung kontaktiert. Sie finden ein detailliertes Setup in dem LDAP-HOWTO (`http://www.tldp.org/HOWTO/LDAP-HOWTO.html`) (`/usr/share/doc/HOWTO/en-txt/LDAP-HOWTO.txt.gz`).

Sie können mehr über NIS-Sicherheit in dem NIS-HOWTO (`http://www.tldp.org/HOWTO/NIS-HOWTO.html`) (`/usr/share/doc/HOWTO/en-txt/NIS-HOWTO.txt.gz`) lesen.

FIXME (jfs): Add info on how to set this up in Debian.

5.13 Sichern von RPC-Diensten

Sie sollten RPC abschalten, wenn Sie es nicht benötigen.

Remote Procedure Call (RPC, etwa »Entfernter Funktionsaufruf«) ist ein Protokoll, das von Programmen verwendet werden kann, um Dienste von anderen Programmen, die auf anderen Computern laufen, anzufordern. Der `portmap`-Dienst kontrolliert RPC-Dienste durch Abbilden von RPC-Programmnummern auf DARPA-Protokoll-Portnummern. Er muss laufen, um RPC-Aufrufe ausführen zu können.

RPC-basierte Dienste hatten eine unrühmliche Geschichte, was Sicherheitslücken betrifft, obwohl dies für den Portmapper an sich nicht gilt (dieser bietet aber nach wie vor entfernten

Angreifern Informationen). Es ist zu beachten, dass einige DDoS-(distributed denial of service, verteilte Dienstverweigerungen)-Angriffe RPC-Löcher benutzen, um in das System einzudringen und als so genannter Agent/Handler zu fungieren.

Sie benötigen RPC nur dann, wenn Sie einen RPC-basierten Dienst verwenden. Die bekanntesten RPC-basierten Dienste sind NFS (Network File System) und NIS (Network Information System). Vergleichen Sie mit dem vorherigen Abschnitt für weitere Information über NIS. Der File Alteration Monitor (FAM), der vom Paket `fam` bereitgestellt wird, ist ebenso ein RPC-Dienst und hängt deshalb von `portmap` ab.

NFS-Dienste sind in einigen Netzwerken ziemlich wichtig. Wenn dies für Sie der Fall ist, müssen Sie einen Ausgleich finden, zwischen Sicherheit und Nutzbarkeit für Ihr Netzwerk. Sie können mehr über NFS-Sicherheit im NFS-HOWTO (`http://www.tldp.org/HOWTO/NFS-HOWTO.html`) (`/usr/share/doc/HOWTO/en-txt/NFS-HOWTO.txt.gz`) finden.

5.13.1 Vollständiges Deaktivieren von RPC-Diensten

Das Abschalten von portmap ist relativ einfach. Es gibt verschiedene Methoden. Die einfachste in einem Debian 3.0 oder neueren System ist das Paket `portmap` zu deinstallieren. Wenn Sie eine ältere Version von Debian laufen haben, werden Sie den Dienst, wie in 'Daemons abschalten' auf Seite 40 beschrieben, abschalten müssen, weil das Programm Teil des Pakets `netbase` (das nicht deinstalliert werden kann, ohne das System kaputt zu machen) ist.

Beachten Sie, dass einige Desktop-Umgebungen (hauptsächlich GNOME) RPC-Dienste verwenden und den Portmapper für einige der Dateimanager-Eigenschaften benötigen. Wenn dies bei Ihnen der Fall ist, können Sie den Zugang zu RPC-Diensten, wie weiter unter beschrieben, beschränken.

5.13.2 Einschränken des Zugriffs auf RPC-Dienste

Unglücklicherweise ist es in manchen Fällen nicht möglich, RPC-Dienste vom System zu entfernen. Einige lokale Desktop-Dienste (hauptsächlich SGIs `fam`) sind RPC-basiert und benötigen deshalb einen lokalen Portmapper. Dies bedeutet, dass unter bestimmten Umständen Benutzer die eine Desktop-Umgebung (wie GNOME) installieren, den Portmapper auch installieren werden.

Es gibt einige Wege den Zugriff auf den Portmapper und RPC-Dienste zu beschränken:

- Blockieren des Zugangs zu den Ports, die von diesen Diensten verwendet werden, mit einer lokalen Firewall (vergleiche 'Hinzufügen von Firewall-Fähigkeiten' auf der nächsten Seite).
- Blockieren des Zugangs zu diesen Diensten mittels TCP-Wrappers, da der Portmapper (und einige RPC-Dienste) mit `libwrap` (siehe 'Die Nutzung von tcpwrappers' auf Seite 73) kompiliert worden. Dies bedeutet, dass Sie Zugang zu diesen durch die `hosts.allow` und `hosts.deny` TCP-Wrapper-Konfiguration blockieren.

- Seit Version 5-5 kann das Paket `portmap` so konfiguriert werden, dass es nur noch an der lokalen Schleifenschnittstelle lauscht. Um dies zu erreichen, kommentieren Sie die folgende Zeile in der Datei `/etc/default/portmap` aus: `#OPTIONS="-i 127.0.0.1"` und starten Sie den Portmapper neu. Dies ist ausreichend, um lokale RPC-Dienste laufen zu lassen, während zur selben Zeit entfernte Systeme am Zugang gehindert werden (lesen Sie dazu auch 'Lösung des Problems der Weak-End-Hosts' auf Seite 92).

5.14 Hinzufügen von Firewall-Fähigkeiten

Das Debian-GNU/Linux-Betriebssystem hat die eingebauten Fähigkeiten des Linux-Kernels.[9] Dies heißt, dass Sie, wenn Sie ein Potato-System (Debian 2.2 mit dem standardmäßigen Kernel 2.2) installiert haben, `ipchains` Firewall-Unterstützung im Kernel haben. Das Paket `ipchains` sollte bereits (aufgrund seiner Priorität) installiert sein. Wenn Sie ein Debian 3.0 (oder 3.1) System installiert haben (der Standard-Kernel ist 2.4), so verfügen Sie auch über die `iptables` (netfilter) Firewall. Der Hauptunterschied zwischen `ipchains` und `iptables` ist, dass letzteres auf *stateful packet inspection* (zustandsbehaftete Paketuntersuchung) beruht, so dass Ihnen sicherere (und einfacher zu erstellende) Filterkonfigurationen zur Verfügung stehen.

5.14.1 Firewallen des lokalen Systems

Sie können eine Firewall dazu benutzen, den Zugriff auf Ihr lokales System abzusichern und sogar um die Kommunikation von ihm nach Außen zu beschränken. Firewall-Regeln können auch dazu benutzt werden, Prozesse zu sichern, die nicht vernünftig konfiguriert werden können, um Dienste *nicht* einigen Netzwerken, IP-Adressen, etc. zur Verfügung zu stellen.

Dieser Schritt ist aber hauptsächlich deshalb als letzter in dieser Anleitung, weil es *viel* besser ist, sich nicht alleine auf die Fähigkeiten der Firewall zu verlassen, um ein System zu schützen. Die Sicherheit eines Systems setzt sich aus mehreren Ebenen zusammen; eine Firewall sollte die letzte sein, wenn bereits alle Dienste abgehärtet worden sind. Sie können sich sicherlich leicht eine Konfiguration vorstellen, bei der ein System lediglich von einer eingebauten Firewall geschützt wird, und der Administrator glückselig die Firewall-Regeln aus irgendwelchen Gründen (Probleme mit dem Setup, Verdruss, Denkfehler, ...) entfernt. Dieses System wäre weit geöffnet für Angriffe, wenn es keine anderen Schutzmaßnahmen auf dem System gibt.

Andererseits können Firewall-Regeln auf dem lokalen System dafür sorgen, dass böse Dinge nicht passieren. Sogar wenn die bereitgestellten Dienste sicher konfiguriert sind, kann eine Firewall vor Misskonfigurationen oder frisch installierten Diensten, die noch nicht passend konfiguriert sind, schützen. Außerdem wird eine strenge Konfiguration *nach Hause telefonierende* Trojaner am Funktionieren hindern, es sei denn, der Firewall-Code wird entfernt. Beachten Sie, dass ein Eindringling *keinen* Superuser-Zugriff benötigt, um ferngesteuerte Trojaner zu installieren (da es erlaubt ist, sich an Ports zu binden, wenn es sich nicht um einen privilegierten Port handelt und die Fähigkeiten (Capabilities) noch vorhanden sind).

[9] Ältere Debian-Distributionen benötigen einen passenden Kernelpatch. Zum Beispiel verwendete Debian 2.1 den Kernel 2.0.34, der diese Funktionen nicht enthielt.

Demzufolge wäre ein passendes Firewall-Setup, eines mit einer standardmäßigen Deny-Policy (was also alles ablehnt, was nicht ausdrücklich erlaubt ist), und weiterhin:

- Eingehende Verbindungen werden nur zu lokalen Diensten von erlaubten Maschinen gestattet.
- Ausgehende Verbindungen werden nur von Diensten erlaubt, die auf Ihrem System benutzt werden (DNS, Web-Surfen, POP, E-Mail, ...).[10]
- Die Forward-Regel verbietet alles; es sei denn, andere Systeme werden geschützt (siehe dazu unten).
- Alle anderen eingehenden und ausgehenden Verbindungen werden abgelehnt.

5.14.2 Schützen anderer Systeme durch eine Firewall

Eine Debian-Firewall kann auch so installiert werden, dass sie mit Firewall-Regeln Systeme *hinter* ihr beschützt, indem sie die Angriffsfläche zum Internet hin einschränkt. Eine Firewall kann so konfiguriert werden, dass ein Zugriff von Systemen außerhalb des lokalen Netzwerks auf interne Dienste (Ports) unterbunden wird. Zum Beispiel muss auf einem Mail-Server lediglich Port 25 (auf dem der Mail-Dienst aufsetzt) von außen zugänglich sein. Eine Firewall kann so konfiguriert werden, dass sogar, wenn es neben den öffentlich zugänglichen noch andere Netzwerkdienste gibt, direkt an diese gesendete Pakete verworfen werden (dies nennt man *filtern*).

Sie können eine Debian GNU/Linux Maschine sogar so konfigurieren, dass sie als Bridge-Firewall (überbrückender Schutzwall) fungiert, d.h. als eine filternde Firewall, die komplett transparent zum gesamten Netzwerk erscheint, ohne IP-Adresse auskommt und daher nicht direkt attackiert werden kann. Abhängig von dem installierten Kernel müssen Sie vielleicht den Bridge-Firewall-Patch installieren und dann *802.1d Ethernet Bridging* in der Kernel-Konfiguration und die neue Option *netfilter (firewalling) Support* auswählen. Sehen Sie dazu 'Aufsetzen einer überbrückenden Firewall (bridge Firewall)' auf Seite 237, um zu erfahren, wie man dies auf einem Debian GNU/Linux System aufsetzt.

5.14.3 Aufsetzen einer Firewall

Die Debian-Standardinstallation bietet im Gegensatz zu vielen anderen Linux-Distributionen noch keine Methode für den Administrator, eine Firewall-Konfiguration mit der Standardinstallation einzurichten, aber Sie können eine Anzahl von Firewall-Konfigurationspaketen (siehe 'Nutzen von Firewall-Paketen' auf der nächsten Seite) installieren.

Natürlich ist die Konfiguration einer Firewall immer vom System und dem Netzwerk abhängig. Ein Administrator muss vorher das Netzwerklayout und die Systeme, die er beschützen

[10] Im Gegensatz zu persönlichen Firewalls für andere Betriebssysteme, stellt Debian GNU/Linux (noch) keine Firewall-Erstellungs-Schnittstelle zur Verfügung, die Regeln erstellen kann, die einzelne Prozesse oder Benutzer einschränken. Jedoch kann der Iptables-Code so konfiguriert werden, dass er dies kann (siehe dazu das "owner"-Modul in der Handbuchseite `iptables(8)`).

will, kennen, die Dienste, auf die zugegriffen werden können muss, und ob andere netzwerkspezifischen Erwägungen (wie NAT oder Routing) berücksichtigt werden müssen. Seien Sie vorsichtig, wenn Sie Ihre Firewall konfigurieren. Wie Laurence J. Lane im `iptables`-Paket sagt:

Die Werkzeuge können leicht falsch verwendet werden und eine Menge Ärger verursachen, indem sie den gesamten Zugang zu einem Computernetzwerk stilllegen. Es ist nicht völlig ungewöhnlich, dass sich ein Systemadministrator, der ein System verwaltet, das hunderte oder tausende von Kilometern entfernt ist, irrtümlicherweise selbst davon ausgeschlossen hat. Man kann es sogar schaffen, sich von dem Computer auszusperren, dessen Tastatur unter seinen Fingern liegt. Lassen Sie daher die gebotene Vorsicht walten.

Vergessen Sie nicht: Das bloße Installieren von `iptables` (oder dem älteren Firewallcode) gibt Ihnen keine Sicherheit, es stellt lediglich die Software zur Verfügung. Um eine Firewall zu haben, müssen Sie sie konfigurieren!

Wenn Sie keine Ahnung haben, wie Sie Ihre Firewall-Regeln manuell aufsetzen sollen, sehen Sie in dem *Packet Filtering HOWTO* und *NAT HOWTO* aus dem Paket `iptables`, zu finden unter `/usr/share/doc/iptables/html/` nach.

Wenn Sie nicht viel über Firewalls wissen, sollten Sie beginnen, indem Sie das Firewalling and Proxy Server HOWTO (`http://www.tldp.org/HOWTO/Firewall-HOWTO.html`) lesen. Installieren Sie das Paket `doc-linux-text`, wenn Sie es offline lesen wollen. Wenn Sie Fragen stellen wollen oder Hilfe beim Einrichten einer Firewall benötigen, können Sie sich an die debian-firewall-Mailingliste wenden, siehe `http://lists.debian.org/debian-firewall`. Sehen Sie auch 'Seien Sie wachsam gegenüber generellen Sicherheitsproblemen!' auf Seite 29 für weitere (allgemeinere) Verweise zu Firewalls. Ein weiterer guter Leitfaden für Iptables ist `http://iptables-tutorial.frozentux.net/iptables-tutorial.html`.

Nutzen von Firewall-Paketen

Das manuelle Aufsetzen einer Firewall kann für neue (und manchmal auch für erfahrene) Administratoren kompliziert sein. Hierfür hat die Freie-Software-Gemeinschaft eine große Zahl von Werkzeugen erstellt, die zur einfachen Konfiguration einer lokalen Firewall benutzt werden können. Seien Sie gewarnt, dass einige dieser Werkzeuge sich mehr auf lokalen Schutz konzentrieren (auch *personal firewall* genannt), während andere vielseitiger sind und dazu benutzt werden können, komplexere Regelwerke zum Schutz ganzer Netzwerke zu erstellen.

Einige Programme, die unter Debian zum Aufsetzen von Firewall-Regeln benutzt werden können, sind:

- `firestarter`, eine GNOME-Anwendung, die sich an Endanwender richtet, die einen Wizard enthält, der nützlich ist, um schnell Firewall-Regeln aufzustellen. Die Anwendung enthält eine graphische Oberfläche zum Beobachten, ob eine Firewall-Regel Daten blockiert.

- `fwbuilder`, eine objektorientierte graphische Oberfläche, die Richtlinien-Compiler für verschiedene Firewall-Plattformen inklusive Linux' netfilter, BSDs pf (verwendet in OpenBSD, NetBSD, FreeBSD und MacOS X) ebenso wie Zugriffslisten von Routern enthält. Es ist ähnlich zu Enterprise-Firewall-Management-Software. Die vollständige Funktionalität von fwbuilder ist auch von der Kommandozeile verfügbar.

- `shorewall`, ein Firewall-Konfigurationswerkzeug, das Unterstützung für IPsec sowie beschränkte Unterstützung für Traffic Shaping und die Definition der Firewall-Regeln bietet. Die Konfiguration geschieht durch eine einfache Menge von Dateien, die verwendet werden, um die iptables-Regeln aufzustellen.

- `guarddog`, ein KDE-basiertes Firewall-Konfigurations-Paket, das sich an Anfänger wie auch an fortgeschrittene Benutzer richtet.

- `knetfilter`, eine KDE-Oberfläche zur Verwaltung von Firewall- und NAT-Regeln für iptables (Alternative/Konkurrent zum guarddog-Werkzeug, obwohl es sich leicht an fortgeschrittene Benutzer richtet).

- `bastille`, diese Härtungsanwendung ist in 'Automatisches Abhärten von Debian-Systemen' auf Seite 133 beschrieben. Einer der Härtungsschritte, die der Administrator konfigurieren kann, ist eine Definition des erlaubten und verbotenen Netzwerkverkehrs, der verwendet wird, eine Anzahl von Firewall-Regeln, die das System am Start ausführt, zu generieren.

Es gibt in Debian auch noch eine Menge anderer Frontends für Iptables. Eine vollständige Liste kann auf der Firewall-Seite des Debian-Wikis (`http://wiki.debian.org/Firewalls`), die auch einen Vergleich der verschiedenen Pakete enthält, abgerufen werden.

Seien Sie gewarnt, dass manche der zuvor skizzierten Pakete Firewall-Skripte einführen, die beim Systemstart ausgeführt werden. Testen Sie diese ausführlich, bevor Sie neustarten, oder Sie finden sich selbst ausgesperrt vor Ihrem Rechner wieder. Wenn Sie verschiedene Firewall-Pakete mischen, kann dies zu unerwünschten Nebeneffekten führen. Gewöhnlich wird das Firewall-Skript, das zuletzt ausgeführt wird, das System konfigurieren (was Sie so vielleicht nicht vorhatten). Sehen Sie hierzu in der Paketdokumentation nach und benutzen Sie nur eines dieser Setups.

Wie bereits zuvor erläutert, sind einige Programme wie `firestarter`, `guarddog` und `knetfilter` graphische Administrations-Schnittstellen, die entweder GNOME oder KDE (die letzte beiden) benutzen. Diese sind viel benutzerorientierter (z.B. für Heimanwender) als einige der anderen Pakete in der Liste, die sich eher an Administratoren richten. Einige der Programme die zuvor aufgeführt wurden (wie `bastille`) fokussieren auf dem Erstellen von Firewall-Regeln zum Schützen des Rechners, auf dem sie laufen, sind aber nicht notwendigerweise dafür geschaffen, Firewall-Regeln für Rechner zu erstellen, die ein Netzwerk schützen (wie `shorewall` oder `fwbuilder`).

Es gibt einen weiteren Typ von Firewall-Anwendungen: Anwendungs-Proxys. Wenn Sie eine Möglichkeit suchen, eine Unternehmenslösung aufzusetzen, die Pakete filtert und eine Anzahl von transparenten Proxys bietet, die feinabgestimmte Verkehrsanalysen bieten, so sollten Sie

`zorp` genauer betrachten. Dies bietet alles in einem einzelnen Programm. Sie können diese Art von Firewall-Rechner auch manuell aufsetzen, indem Sie die Proxys, die in Debian vorhanden sind, für verschiedene Dienste verwenden. Zum Beispiel für DNS `bind` (richtig konfiguriert), `dnsmasq`, `pdnsd` oder `totd` für FTP `frox` oder `ftp-proxy`, für X11 `xfwp`, für IMAP `imapproxy`, für Mail `smtpd` oder für POP3 `p3scan`. Für andere Protokolle können Sie entweder einen allgemeinen TCP-Proxy wie `simpleproxy` oder einen allgemeinen SOCKS-Proxy wie `dante-server`, `tsocks` oder `socks4-server` verwenden. Typischerweise werden Sie auch ein Web-Cache-System (wie `squid`) und ein Web-Filtersystem (wie `squidguard` oder `dansguardian`) nutzen.

Manuelle init.d-Konfiguration

Eine andere Möglichkeit ist die manuelle Konfiguration Ihrer Firewall-Regeln durch ein init.d-Skript, das die `iptables`-Befehle ausführt. Befolgen Sie diese Schritte:

- Sehen Sie das unten aufgeführte Skript durch und passen Sie es Ihren Anforderungen an.
- Testen Sie das Skript und überprüfen Sie die Syslog-Meldungen nach unterdrückten Netzverkehr. Wenn Sie vom Netzwerk aus testen, werden Sie entweder den Beispielshellcode starten wollen, um die Firewall zu entfernen (wenn Sie nichts innerhalb von 20 Sekunden eingeben) oder Sie sollten die *default deny*-Richtliniendefinition auskommentieren (*-P INPUT DROP* und *-P OUTPUT DROP*) und überprüfen, dass das System keine gültigen Daten verworfen hat.
- Verschieben Sie das Skript nach `/etc/init.d/meineFirewall`
- Konfigurieren Sie das System das Skript zu starten, bevor irgendein Netzwerk konfiguriert wird:

```
#update-rc.d meineFirewall start 40 S . stop 89 0 6 .
```

Dies ist das Beispiel-Firewallskript:

```
#!/bin/sh
# Simple example firewall configuration.
#
# Caveats:
# - This configuration applies to all network interfaces
#   if you want to restrict this to only a given interface use
#   '-i INTERFACE' in the iptables calls.
# - Remote access for TCP/UDP services is granted to any host,
#   you probably want to restrict this using '--source'.
#
# chkconfig: 2345 9 91
# description: Activates/Deactivates the firewall at boot time
```

```
#
# You can test this script before applying with the following shell
# snippet, if you do not type anything in 10 seconds the firewall
# rules will be cleared.
#-----------------------------------------------------------------
#  while true; do test=""; read  -t 20 -p "OK? " test ; \
#  [ -z "$test" ] && /etc/init.d/meineFirewall clear ; done
#-----------------------------------------------------------------

PATH=/bin:/sbin:/usr/bin:/usr/sbin

# Services that the system will offer to the network
TCP_SERVICES="22" # SSH only
UDP_SERVICES=""
# Services the system will use from the network
REMOTE_TCP_SERVICES="80" # web browsing
REMOTE_UDP_SERVICES="53" # DNS
# Network that will be used for remote mgmt
# (if undefined, no rules will be setup)
# NETWORK_MGMT=192.168.0.0/24
# Port used for the SSH service, define this is you have setup a
# management network but remove it from TCP_SERVICES
# SSH_PORT="22"

if ! [ -x /sbin/iptables ]; then
    exit 0
fi

fw_start () {

  # Input traffic:
  /sbin/iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
  # Services
  if [ -n "$TCP_SERVICES" ] ; then
  for PORT in $TCP_SERVICES; do
    /sbin/iptables -A INPUT -p tcp --dport ${PORT} -j ACCEPT
  done
  fi
  if [ -n "$UDP_SERVICES" ] ; then
  for PORT in $UDP_SERVICES; do
    /sbin/iptables -A INPUT -p udp --dport ${PORT} -j ACCEPT
  done
  fi
  # Remote management
  if [ -n "$NETWORK_MGMT" ] ; then
    /sbin/iptables -A INPUT -p tcp --src ${NETWORK_MGMT} --dport ${SSH_PORT}
```

```
else
  /sbin/iptables -A INPUT -p tcp --dport ${SSH_PORT}  -j ACCEPT
fi
# Remote testing
/sbin/iptables -A INPUT -p icmp -j ACCEPT
/sbin/iptables -A INPUT -i lo -j ACCEPT
/sbin/iptables -P INPUT DROP
/sbin/iptables -A INPUT -j LOG

# Output:
/sbin/iptables -A OUTPUT -j ACCEPT -o lo
/sbin/iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# ICMP is permitted:
/sbin/iptables -A OUTPUT -p icmp -j ACCEPT
# So are security package updates:
# Note: You can hardcode the IP address here to prevent DNS spoofing
# and to setup the rules even if DNS does not work but then you
# will not "see" IP changes for this service:
/sbin/iptables -A OUTPUT -p tcp -d security.debian.org --dport 80 -j ACCEPT
# As well as the services we have defined:
if [ -n "$REMOTE_TCP_SERVICES" ] ; then
for PORT in $REMOTE_TCP_SERVICES; do
  /sbin/iptables -A OUTPUT -p tcp --dport ${PORT} -j ACCEPT
done
fi
if [ -n "$REMOTE_UDP_SERVICES" ] ; then
for PORT in $REMOTE_UDP_SERVICES; do
  /sbin/iptables -A OUTPUT -p udp --dport ${PORT} -j ACCEPT
done
fi
# All other connections are registered in syslog
/sbin/iptables -A OUTPUT -j LOG
/sbin/iptables -A OUTPUT -j REJECT
/sbin/iptables -P OUTPUT DROP
# Other network protections
# (some will only work with some kernel versions)
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 0 > /proc/sys/net/ipv4/ip_forward
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
echo 1 > /proc/sys/net/ipv4/conf/all/log_martians
echo 1 > /proc/sys/net/ipv4/ip_always_defrag
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter
echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects
echo 0 > /proc/sys/net/ipv4/conf/all/accept_source_route
```

```
}

fw_stop () {
  /sbin/iptables -F
  /sbin/iptables -t nat -F
  /sbin/iptables -t mangle -F
  /sbin/iptables -P INPUT DROP
  /sbin/iptables -P FORWARD DROP
  /sbin/iptables -P OUTPUT ACCEPT
}

fw_clear () {
  /sbin/iptables -F
  /sbin/iptables -t nat -F
  /sbin/iptables -t mangle -F
  /sbin/iptables -P INPUT ACCEPT
  /sbin/iptables -P FORWARD ACCEPT
  /sbin/iptables -P OUTPUT ACCEPT
}


case "$1" in
  start|restart)
    echo -n "Starting firewall.."
    fw_stop
    fw_start
    echo "done."
    ;;
  stop)
    echo -n "Stopping firewall.."
    fw_stop
    echo "done."
    ;;
  clear)
    echo -n "Clearing firewall rules.."
    fw_clear
    echo "done."
    ;;
  *)
    echo "Usage: $0 {start|stop|restart|clear}"
    exit 1
    ;;
  esac
exit 0
```

Konfiguration von Firewall-Regeln mittels ifup

Sie können auch die Netzwerkkonfiguration in `/etc/network/interfaces` verwenden, um Ihre Firewall-Regeln einzurichten. Dafür müssen Sie Folgendes tun:

- Erstellen Sie Ihre Firewall-Regeln für die aktivierte Schnittstelle.
- Sichern Sie Ihre Regeln mit `iptables-save` in eine Datei in `/etc`, zum Beispiel `/etc/iptables.up.rules`
- Konfigurieren Sie `/etc/network/interfaces`, diese Regeln zu verwenden:

```
iface eth0 inet static
        address x.x.x.x
        [.. interface configuration ..]
        pre-up iptables-restore < /etc/iptables.up.rules
```

Wahlweise können Sie auch Regeln erstellen, die beim Herunterfahren der Netzwerkschnittstelle ausgeführt werden. Dazu erzeugen Sie diese, speichern sie in `/etc/iptables.down.rules` und fügen diese Anweisung zur Schnittstellenkonfiguration hinzu:

```
post-down iptables-restore < /etc/iptables.down.rules
```

Für weitergehende Firewall-Konfigurationsskripte durch `ifupdown` können Sie die zu jeder Schnittstelle verfügbaren Hooks (Einspringpunkte) wie in den `*.d/`-Verzeichnissen verwenden, die mit `run-parts` aufgerufen werden (vergleiche `run-parts(8)`).

Machen Sie es auf die (alte) Debian-Art

ACHTUNG: Diese Informationen gelten nur für iptables in *Woody*. Versionen nach 1.2.7-8 haben nicht mehr länger das init.d-Skript, das hier beschrieben wird. Benutzer von Debian 3.1 (*Sarge*) oder neueren Veröffentlichungen sollten entweder die Firewall-Regeln manuell aufbauen oder eines der Programme zur Erzeugung einer Firewall verwenden, die weiter oben beschrieben wurden.

Wenn Sie Debian 3.0 oder neuer benutzen, werden Sie feststellen, dass Sie das Paket `iptables` installiert haben. Dies ist die Unterstützung für die Netfilter-Implementation in 2.4.4+ Kerneln. Da das System nach der Installation aber keine Firewall-Regeln kennen kann (Firewall-Regeln sind zu systemspezifisch), müssen Sie iptables einschalten. Wie auch immer: Die Skripte wurden so konfiguriert, dass der Administrator Firewall-Regeln aufsetzen kann und die init-Skripte sie dann *lernen* können und so immer als das Setup der Firewall fungieren.

Hierzu müssen Sie Folgendes tun:

- Konfigurieren Sie das Paket so, dass es mit dem System gestartet wird. Bei neueren Versionen (seit 1.2.6a-1) werden Sie während der Installation hiernach gefragt. Sie können es hinterher wieder mit `dpkg-reconfigure -plow iptables` ändern. *Wichtig*: Bei älteren Versionen geschah dies noch durch Editieren von `/etc/default/iptables`, so dass die Variable `enable_iptables_initd` auf *true* gesetzt wurde.
- Erstellen Sie ein Firewall-Setup mit iptables, benutzen Sie dazu die Kommandozeile (siehe `iptables(8)`) oder andere Werkzeuge aus Debians Firewall-Paketen (siehe 'Nutzen von Firewall-Paketen' auf Seite 124). Sie müssen einen Satz von Firewall-Regeln erstellen, die benutzt werden sollen, wenn die Firewall *aktiv* ist, und einen anderen, wenn die Firewall *inaktiv* (dies können auch nur leere Regeln sein) ist.
- Sichern Sie die erstellten Regeln mit den Skripten `/etc/init.d/iptables save active` und `/etc/init.d/iptables save inactive`.

Sobald dies geschehen ist, ist Ihr Firewall-Setup im Verzeichnis `/var/lib/iptables/` gespeichert und wird beim System-Boot ausgeführt (oder wenn das initd-Skript mit *start*- oder *stop*-Argument gestartet wird). Beachten Sie, dass die standardmäßigen Einstellungen unter Debian vorsehen, den Firewall-Code in den Multiuser-Runleveln (2 bis 5) sehr früh (10) zu starten. Außerdem wird er im singleuser-Runlevel (1) gestoppt. Ändern Sie dies, wenn es nicht Ihren lokalen Richtlinien entspricht.

Bitte lesen Sie die Kommentare in der `/etc/default/iptables`-Konfigurationsdatei für weitere Informationen zu den dieses Paket betreffenden Dingen.

Testen Ihrer Firewall-Konfiguration

Testen Ihrer Firewall-Konfiguration ist so einfach und so schwierig, wie das Starten Ihres Firewall-Skripts (oder die Aktivierung der Konfiguration, die Sie in Ihrer Firewall-Konfigurationsanwendung definierten). Wenn Sie jedoch nicht sorgfältig genug sind und Sie Ihre Firewall aus der Ferne konfigurieren (z.B. durch eine SSH-Verbindung), könnten Sie sich selbst aussperren.

Es gibt mehrere Möglichkeiten, dies zu verhindern. Eine ist das Starten eines Skriptes in einem separaten Terminal, das Ihre Firewall-Konfiguration entfernt, wenn es keine Eingabe von Ihnen erhält. Ein Beispiel dafür ist:

```
$  while true; do test=""; read  -t 20 -p "OK? " test ; \
  [ -z "$test" ] && /etc/init.d/firewall clear ; done
```

Eine andere Möglichkeit ist das Einführen einer Hintertür in Ihr System durch einen alternativen Mechanismus, der es Ihnen erlaubt, das Firewall-System entweder zurückzusetzen oder ein Loch in es schlägt, wenn irgendetwas krumm läuft. Dafür können Sie `knockd` verwenden und es so konfigurieren, dass eine spezielle Portverbindungsversuchssequenz die Firewall zurücksetzt (oder eine temporäre Regel hinzufügt). Selbst wenn die Pakete von der Firewall zurückgewiesen werden, werden Sie Ihr Problem lösen können, da `knockd` auf der Schnittstelle lauscht und Sie *sieht*.

Das Testen einer Firewall, die ein internen Netz schützt, ist eine andere Aufgabe. Schauen Sie sich dafür einige Werkzeuge an, die es für entfernte Ausnutzbarkeitsbewertungen gibt (siehe 'Programme zur Fernprüfung der Verwundbarkeit' auf Seite 163), um das Netzwerk von außerhalb nach innen (oder aus einer beliebig anderen Richtung) bezüglich der Effektivität der Firewall-Konfiguration zu testen.

Kapitel 6

Automatisches Abhärten von Debian-Systemen

Nachdem Sie nun all die Informationen aus den vorherigen Kapiteln gelesen haben, fragen Sie sich vielleicht: "Ich habe sehr viele Dinge zu erledigen, um mein System abzusichern. Könnte man das nicht automatisieren?" Die Antwort lautet: "Ja, aber seien Sie vorsichtig mit automatischen Werkzeugen." Manche Leute denken, dass ein Absicherungswerkzeug nicht die Notwendigkeit einer guten Systemadministration abschafft. Täuschen Sie sich also nicht selbst, indem Sie denken, dass Sie all die Prozesse automatisieren könnten, und sich alle betreffenden Angelegenheiten von selbst erledigen würden. Sicherheit ist ein andauernder Prozess, an dem der Administrator teilnehmen muss. Er kann nicht einfach wegbleiben und irgendwelche Werkzeuge die Arbeit erledigen lassen, weil kein einzelnes Werkzeug die Umsetzung aller Sicherheitsrichtlinien, aller Angriffe oder aller Umgebungen bewältigen kann.

Seit Woody (Debian 3.0) gibt es zwei unterschiedliche Pakete, die zur Erhöhung der Sicherheit nützlich sind. Das Paket `harden` versucht auf Basis der Paket-Abhängigkeiten schnell, wertvolle Sicherheitspakete zu installieren und Pakete mit Mängeln zu entfernen. Die Konfiguration der Pakete muss der Administrator erledigen. Das Paket `bastille` implementiert gegebene Sicherheitsregeln für das lokale System, die auf einer vorhergehenden Konfiguration durch den Administrator basieren (Sie können auch mit einfachen Ja/Nein-Fragen durch die Konfiguration geführt werden).

6.1 Harden

Das Paket `harden` versucht es einfacher zu machen, Rechner, die gute Sicherheit benötigen, zu installieren und zu administrieren. Dieses Paket sollte von Leuten benutzt werden, die eine schnelle Hilfe bei der Erhöhung der Systemsicherheit haben wollen. Es installiert automatisch einige Werkzeuge, die die Sicherheit auf unterschiedliche Art und Weise erhöhen: Werkzeuge zur Eindringlingserkennung, Werkzeuge zur Sicherheitsanalyse und mehr. `harden` installiert die folgenden *virtuellen* Pakete (d.h. sie enthalten nichts, hängen aber von anderen Paketen ab oder empfehlen diese):

- `harden-tools`: Werkzeuge, die die Sicherheit des Systems erhöhen (Integritätsprüfung, Eindringlingserkennung, Kernel-Patches, ...)
- `harden-environment`: Hilft eine abgesicherte Umgebung zu konfigurieren (derzeit leer)
- `harden-servers`: entfernt Server, die aus irgendeinem Grund als unsicher gelten
- `harden-clients`: entfernt Clients, die aus irgendeinem Grund als unsicher gelten
- `harden-remoteaudit`: Werkzeuge, um Systeme aus der Ferne zu überprüfen.
- `harden-nids`: hilft bei der Installation eines Systems zur Entdeckung von Netzwerkeindringlingen.
- `harden-surveillance`: hilft bei der Installation von Werkzeugen zum Überwachen von Netzwerken und Diensten.

Nützliche Pakete, für die keine Abhängigkeit besteht:

- `harden-doc`: Stellt dieses und andere sicherheitsrelevante Dokumente zur Verfügung.
- `harden-development`: Entwicklungswerkzeuge, um sicherere Programme zu erstellen.

Seien Sie vorsichtig, wenn Sie Software installiert haben, die Sie brauchen (und aus bestimmten Gründen nicht deinstallieren wollen), und die aufgrund eines Konflikts mit einem der oben aufgeführten Pakete nicht installiert werden kann. In diesem Fall können Sie `harden` nicht vollständig nutzen.

Die harden Pakete machen eigentlich gar nichts. Zumindest nicht unmittelbar. Sie haben jedoch absichtliche Paketkonflikte mit bekannten, unsicheren Paketen. Auf diese Art wird die Paketverwaltung von Debian die Installation dieser Paketen nicht erlauben. Wenn Sie zum Beispiel bei installiertem `harden-servers`-Paket versuchen, einen telnet-Daemon zu installieren, wird Ihnen `apt` Folgendes sagen:

```
# apt-get install telnetd
Die folgenden Pakete werden ENTFERNT:
  harden-servers
Die folgenden NEUEN Pakete werden installiert:
  telnetd
Möchten Sie fortfahren? [J/n]
```

Dies sollte im Kopf des Administrators eine Alarmglocke auslösen, der sein Vorgehen überdenken sollte.

6.2 Bastille Linux

Bastille Linux (http://www.bastille-linux.org) ist ein Werkzeug zur automatischen Abhärtung, das ursprünglich für die Linux-Distributionen Red Hat und Mandrake gedacht war. Wie auch immer: Das Paket `bastille` aus Debian (seit Woody) ist durch Patches angepasst, um dieselbe Funktionalität unter Debian GNU/Linux Systemen zur Verfügung zu stellen.

Bastille kann mit verschiedenen Oberflächen bedient werden (alle sind in ihrem eigenen Handbuch dokumentiert), die dem Administrator erlauben:

- Schritt für Schritt Fragen zur erwünschten Sicherheit Ihres Systems zu beantworten (siehe `InteractiveBastille(8)`),
- Standardeinstellungen zur Sicherheit (zwischen locker, moderat und paranoid) für eine bestimmte Einrichtung (Server oder Arbeitsplatz-Rechner) zu benutzen, und Bastille entscheiden zu lassen, welche Sicherheitsregelungen eingeführt werden sollen (siehe `BastilleChooser(8)`),
- eine vorgefertigte Konfigurationsdatei (von Bastille oder vom Administrator) zu nehmen und eine vorgegebene Sicherheitsregelung zu benutzen (siehe `AutomatedBastille(8)`).

Kapitel 7

Die Infrastruktur für Sicherheit in Debian

7.1 Das Sicherheitsteam von Debian

Debian hat ein Sicherheitsteam, das aus fünf Mitgliedern und zwei Sekretären besteht. Es ist für die Sicherheit in der *Stable*-Veröffentlichung verantwortlich. Das bedeutet, dass es Sicherheitslücken nachgeht, die in Software auftauchen (indem es Foren wie Bugtraq oder vuln-dev beobachtet), und ermittelt, ob davon die *Stable*-Veröffentlichung betroffen ist.

Das Sicherheitsteam von Debian ist auch der Ansprechpartner für Probleme, die von den Programmautoren oder Organisationen wie CERT (`http://www.cert.org`) behandelt werden und die mehrere Linux-Anbieter betreffen können. Das gilt für alle Probleme, die nicht debianspezifisch sind. Es gibt zwei Möglichkeiten, um mit dem Sicherheitsteam in Verbindung zu treten:

- team@security.debian.org (`mailto:team@security.debian.org`), die nur die Mitglieder des Sicherheitsteams lesen.
- security@debian.org (`mailto:security@debian.org`), die von allen Debian-Entwicklern gelesen wird (einschließlich des Sicherheitsteams). E-Mails, die an diese Liste geschickt werden, werden nicht im Internet veröffentlicht (es handelt sich also nicht um eine öffentliche Mailingliste).

Heikle Informationen sollten an die erste Adresse geschickt werden und unter Umständen mit dem Schlüssel von Debian Security Contact (Schlüssel-ID 363CCD95) verschlüsselt werden.

Wenn das Sicherheitsteam ein mögliches Problem erhält, wird es untersuchen, ob die *Stable*-Veröffentlichung davon betroffen ist. Wenn dies der Fall ist, wird eine Ausbesserungen des Quellcodes vorgenommen. Diese Ausbesserung schließt manchmal ein, dass Patches der Programmautoren zurückportiert werden (da das Originalprogramm gewöhnlich eine Versionen weiter ist als das in Debian). Nachdem die Ausbesserung getestet wurde, werden neue Pakete

vorbereitet und auf der Seite `security.debian.org` veröffentlicht, damit sie mit `apt` abgerufen werden können (siehe 'Ausführen von Sicherheitsupdates' auf Seite 48). Zur gleichen Zeit wird eine *Debian-Sicherheits-Ankündigung* (DSA) auf der Webseite veröffentlicht und an öffentliche Mailinglisten einschließlich debian-security-announce (`http://lists.debian.org/debian-security-announce`) und Bugtraq geschickt.

Einige andere häufige Fragen zum Sicherheitsteam von Debian können unter 'Fragen zum Sicherheitsteam von Debian' auf Seite 217 gefunden werden.

7.2 Debian-Sicherheits-Ankündigungen

Debian-Sicherheits-Ankündigungen (DSA) werden erstellt, sobald eine Sicherheitslücke entdeckt wird, die ein Debian-Paket berührt. Diese Anweisungen, die von einem Mitglied des Sicherheitsteams signiert sind, enthalten Informationen zu den betroffenen Versionen und den Orten der Aktualisierungen und ihrer MD5-Summen. Die Informationen sind:

- Versionsnummer der Ausbesserung.
- Art des Problems.
- Ob es aus der Ferne oder lokal ausnutzbar ist.
- Kurze Beschreibung des Pakets.
- Beschreibung des Problems.
- Beschreibung des Exploits.
- Beschreibung der Ausbesserung.

DSAs werden sowohl auf der Hauptseite von Debian (`http://www.de.debian.org/`) als auch auf den Sicherheitsseiten von Debian (`http://www.debian.org/security/`) veröffentlicht. Das passiert normalerweise nicht bis die Website neu erstellt wurde (alle vier Stunden). Daher könnten sie nicht sofort vorhanden sein. Somit ist die vorzugswürdige Informationsquelle die Mailingliste debian-security-announce.

DSAs, die auf der Webseite veröffentlicht wurden, können aktualisiert werden, nachdem sie an öffentliche Mailinglisten verschickt wurden. Eine typische Aktualisierung ist, einen Querverweis auf Datenbanken mit Sicherheitslücken hinzuzufügen. Auch Übersetzungen der DSAs [1] werden nicht an die Sicherheitsmailinglisten geschickt, sondern sind direkt auf der Webseite enthalten.

[1] Übersetzungen sind in bis zu zehn verschiedenen Sprachen verfügbar.

7.2.1 Querverweise der Verwundbarkeiten

Debian stellt eine vollständige Tabelle mit Querverweisen (http://www.de.debian.org/security/crossreferences) zur Verfügung, die alle verfügbaren Verweise für die Anweisungen seit 1998 enthält. Diese Tabelle soll die Verweisübersicht von CVE (http://cve.mitre.org/cve/refs/refmap/source-DEBIAN.html) ergänzen.

Sie werden bemerkt haben, dass die Tabelle Verweise auf Sicherheitsdatenbanken wie Bugtraq (http://www.securityfocus.com/bid), CERT/CC Anweisungen (http://www.cert.org/advisories/) und US-CERT Vulnerability Notes Database (http://www.kb.cert.org/vuls) und auf die CVE-Bezeichnungen (siehe unten) enthält. Diese Verweise werden zur Nutzerfreundlichkeit angeboten, aber nur der CVE-Verweise werden regelmäßig überprüft und eingefügt. Dieses Feature wurde im Juni 2002 der Webseite hinzugefügt.

Das Hinzufügen von Querverweisen auf diese Sicherheitsdatenbanken hat folgende Vorteile:

- Es erleichtert Benutzern von Debian zu erkennen und nachzuvollziehen, welche allgemeinen (veröffentlichten) Anweisungen schon von Debian abgedeckt wurden.
- Systemadministratoren können mehr über die Verwundbarkeit und ihre Auswirkungen lernen, wenn sie den Querverweisen folgen.
- Diese Informationen können benutzt werden, um Ausgaben von Verwundbarkeitsscannern, die Verweise auf CVE enthalten, zu überprüfen, um falsche Positivmeldungen auszusortieren (vergleichen Sie 'Der Scanner X zur Einschätzung der Verwundbarkeit sagt, dass mein Debian-System verwundbar wäre?' auf Seite 213).

7.2.2 CVE-Kompatibilität

Debians Sicherheitsankündigungen wurden am 24. Februar 2004 CVE-kompatibel erklärt (http://www.de.debian.org/security/CVE-certificate.jpg) [2].

Die Entwickler von Debian verstehen die Notwendigkeit, genaue und aktuelle Informationen über den Lage der Sicherheit in der Debian-Distribution zur Verfügung zu stellen. Dies ermöglicht es den Benutzern, mit den Risiken durch neue Sicherheitslücken umzugehen. CVE versetzt uns in die Lage, standardisierte Verweise anzubieten, die es Nutzern ermöglicht, einen Prozess zur Verwaltung der Sicherheit auf Grundlage von CVE (http://www.cve.mitre.org/compatible/enterprise.html) zu entwickeln.

Das Projekt Common Vulnerabilities and Exposures (CVE) (http://cve.mitre.org) wird von der MITRE Corporation betreut und stellt eine Liste von standardisierten Bezeichnungen für Verwundbarkeiten und Sicherheitslücken zur Verfügung.

Debian ist überzeugt, dass es außerordentlich wichtig ist, die Nutzer mit zusätzlichen Informationen im Zusammenhang mit Sicherheitsproblemen, die die Debian-Distribution betreffen,

[2] Der vollständige Fragebogen zur Kompatibilität (http://cve.mitre.org/compatible/phase2/SPI_Debian.html) ist bei CVE erhältlich.

zu versorgen. Indem CVE-Bezeichnungen in den Anweisungen enthalten sind, können Nutzer leichter allgemeine Verwundbarkeiten mit bestimmten Aktualisierungen von Debian in Verbindung bringen. Dies verringert die Zeit, die benötigt wird, um Verwundbarkeiten, die unsere Nutzer betreffen, abzuarbeiten. Außerdem vereinfacht es die Organisation der Sicherheit in einer Umgebung, in der schon Sicherheitswerkzeuge, die CVE verwenden, wie Erkennungssysteme von Eindringlingen in Netzwerk oder Host oder Werkzeuge zur Bewertung der Sicherheit eingesetzt werden, unabhängig davon, ob sie auf der Debian-Distribution beruhen.

Debian begann im Juni 2002, CVE-Bezeichnung zu den DSAs hinzuzufügen. Jetzt sind CVE-Bezeichnungen in allen DSAs seit September 1998 enthalten, nachdem die Nachprüfungsphase im August 2002 begonnen wurde. Alle Anweisungen können auf der Webseite von Debian abgerufen werden. Auch Ankündigungen von neuen Verwundbarkeiten enthalten CVE-Bezeichnungen, wenn sie zum Zeitpunkt ihrer Veröffentlichung verfügbar waren. Anweisungen, die mit einer bestimmten CVE-Bezeichnung verbunden sind, können direkt über die Suchmaschine (`http://search.debian.org/`) gesucht werden.

Benutzer, die nach einer bestimmten CVE-Bezeichnung suchen wollen, können auch die Suchmaschine verwenden, die auf debian.org verfügbar ist, um die verfügbaren Anweisungen (auf Englisch und Übersetzungen in andere Sprachen), die mit den CVE-Bezeichnungen verbunden sind, abzurufen. Eine Suche kann nach einem bestimmten Begriff (z.B. nach der Anweisung CAN-2002-0001 (`http://search.debian.org/?q=advisory+%22CAN-2002-0001%22ps=50o=1m=all`)) oder nach einem Teilbegriff (z.B. alle Kandidaten von 2002, die in Anweisungen enthalten sind, finden Sie mit der Suche nach CAN-2002 (`http://search.debian.org/?q=advisory+%22CAN-2002%22ps=50o=1m=all`)) durchgeführt werden. Beachten Sie, dass Sie das Wort “advisory” zusammen mit der CVE-Bezeichnung eingeben müssen, um nur die Sicherheitsankündigungen zu erhalten.

In einige Fällen finden Sie eine bestimmte CVE-Bezeichnung in veröffentlichten Anweisungen nicht. Beispiele dafür sind:

- Keine Produkte von Debian sind von der Verwundbarkeit betroffen.
- Es gibt noch keine Anweisung, die die Verwundbarkeit abdeckt. Das Sicherheitsproblem wurde vielleicht als Sicherheitsfehler (`http://bugs.debian.org/cgi-bin/pkgreport.cgi?tag=security`) gemeldet, aber eine Ausbesserung wurde noch nicht getestet und hochgeladen.
- Eine Anweisung wurde veröffentlicht, bevor eine CVE-Bezeichnung einer bestimmten Verwundbarkeit zugewiesen wurde (sehen Sie auf der Webseite nach einer Aktualisierung).

7.3 Die Infrastruktur der Sicherheit bei der Paketerstellung in Debian

Da Debian im Moment eine große Anzahl von Architekturen unterstützt, fragen Administratoren manchmal, ob es bei einer bestimmten Architektur bis zu einer Sicherheitsaktualisierung

länger dauert als bei einer anderen. Tatsächlich sind Aktualisierungen auf allen Architekturen zur selben Zeit verfügbar, abgesehen von seltenen Umständen.

Während früher die Sicherheitsaktualisierungen von Hand erstellt wurden, so gilt das heute nicht mehr, wie Anthony Towns in einer Mail (`http://lists.debian.org/debian-devel-announce/2002/debian-devel-announce-200206/msg00002.html`) beschreibt, die am 8. Juni 2002 an die Mailingliste debian-devel-announce geschickt wurde.

Pakete, die vom Sicherheitsteam mit einem passenden Patch (auf `security.debian.org:/org/security.debian.org/queue/unchecked` oder `ftp://security.debian.org/pub/SecurityUploadQueue`) hochgeladen werden, werden innerhalb von 15 Minuten nach dem Hochladen auf Signaturen überprüft. Danach werden sie zu der Liste der Autobuilder hinzugefügt (diese führen nicht mehr einen tägliche Durchgang durch das Archiv durch). Dadurch können die Pakete automatisch für *alle* Architekturen 30 Minuten oder eine Stunde oder so nach dem Hochladen erstellt werden. Allerdings werden Sicherheitsaktualisierungen etwas anderes behandelt als normale Aktualisierungen, die von den Paketbetreuern vorgenommen werden, da in manchen Fällen vor einer Veröffentlichung die Aktualisierungen nochmals getestet werden müssen, eine Anweisung geschrieben werden muss oder eine Woche oder mehr gewartet werden muss, um zu verhindern, dass der Fehler veröffentlicht wird, bevor nicht alle Linux-Anbieter eine vernünftige Chance hatten, ihn zu beheben.

Folglich arbeitet das Archiv der Sicherheitsuploads nach dem folgenden Ablauf (dieser wird *"Accepted-Autobuilding"* genannt):

- Jemand findet ein Sicherheitsproblem.
- Jemand löst das Problem und lädt die Lösung in den Eingang von security.debian.org hoch (dieser *jemand* ist normalerweise ein Mitglied des Sicherheitsteams, kann aber auch ein Paketbetreuer mit einer passenden Verbesserung sein, der sich zuvor mit dem Sicherheitsteam in Verbindung gesetzt hat). Die Änderungsübersicht (changelog) beinhaltet ein *testing-security* oder *stable-security* als Zieldistribution.
- Die hochgeladenen Dateien werden von einem Debian-System überprüft, verarbeitet und in die Warteschleife der angenommenen Dateien weitergeleitet. Danach werden die Buildds benachrichtigt. Auf die Dateien in der Warteschleife kann das Sicherheitsteam und (auf indirektem Wege) die Buildds zugreifen.
- Buildds, die Sicherheit unterstützen, holen sich das Quellpaket (mit einer höheren Priorität als normale Paketerstellungen), erstellen Pakete und schicken die Logs ans Sicherheitsteam.
- Das Sicherheitsteam antwortet auf die Logs und die neu erstellten Pakete werden in die Warteschleife der ungeprüften Dateien hochgeladen, wo sie von einem Debian-System verarbeitet und in die Warteschleife der angenommenen Dateien verschoben werden.
- Wenn das Sicherheitsteam ein Quellpaket akzeptiert (d.h. dass es für alle Architekturen korrekt Pakete erstellt, und dass es die Sicherheitslücke schließt und keine neuen Probleme hervorruft), führen sie ein Skript aus, das

- das Paket im Sicherheitsarchiv installiert,
- die `Paket-`, `Quell-` und `Veröffentlichungsdateien` von security.debian.org auf dem gewöhnlichen Weg aktualisiert (`dpkg-scanpackages`, `dpkg-scansources`, ...),
- eine Vorlage einer Anweisung erstellt, die das Sicherheitsteam fertig stellen kann und
- (wahlweise) die Pakete zu den vorgeschlagenen Aktualisierungen weiterleitet, so dass sie sobald wie möglich in die echten Archive eingefügt werden können.

Dieser Ablauf, der früher per Hand durchgeführt wurde, wurde während des Freezing-Abschnitts von Debian 3.0 Woody (Juli 2002) getestet und umgesetzt. Dank dieser Infrastruktur war es dem Sicherheitsteam möglich, aktualisierte Pakete für Apache- und OpenSSH-Probleme für alle unterstützen Architekturen (fast 20) in weniger als einem Tag bereitzustellen.

7.3.1 Leitfaden über Sicherheitsaktualisierungen für Entwickler

Diese Mail wurde von Wichert Akkerman an die Mailingliste debian-devel-announce (`http://lists.debian.org/debian-devel-announce/2002/debian-devel-announce-200206/msg00004.html`) geschickt, um zu beschreiben, wie Entwickler von Debian Sicherheitsprobleme in ihren Paketen handhaben. Sie wird hier veröffentlicht, sowohl um Entwicklern zu helfen als auch um Nutzern zu verdeutlichen, wie mit Sicherheit in Debian umgegangen wird.

FIXME: Beachten Sie, dass die aktuelle Referenz für diese Informationen die Debian-Entwicklerreferenz (`http://www.debian.org/doc/manuals/developers-reference/ch-pkgs#s-bug-security`) ist und dieser Abschnitt demnächst entfernt wird.

Zusammenarbeit mit dem Sicherheitsteam

Wenn ein Entwickler von einem Sicherheitsproblem erfährt, egal ob in seinem Paket oder in einem anderen, sollte er das immer dem Sicherheitsteam melden (unter team@security.debian.org). Sie gehen ungelösten Sicherheitsproblemen nach, können Paketbetreuern mit Sicherheitsproblemen helfen oder sie selber lösen, sind für den Versand von Sicherheitsankündigungen verantwortlich und betreuen security.debian.org.

Beachten Sie bitte, dass Sicherheitsankündigungen nur für veröffentlichte Distributionen erteilt werden, nicht für Testing, Unstable (siehe 'Wie wird die Sicherheit für Testing und Unstable gehandhabt?' auf Seite 219) und ältere Distributionen (siehe 'Ich verwende eine ältere Version von Debian. Wird sie vom Debian-Sicherheitsteam unterstützt?' auf Seite 219).

Erkennen von Sicherheitsproblemen

Es gibt einige Möglichkeiten, wie ein Entwickler von Sicherheitsproblemen erfahren kann:

- Er bemerkt sie in einem öffentlichem Forum (Mailingliste, Webseite, etc.).
- Jemand reicht einen Fehlerbericht ein. Es sollte dann das *Security*-Tag verwendet oder vom Entwickler hinzugefügt werden.
- Jemand informiert ihn in einer privaten E-Mail.

In den ersten beiden Fällen ist die Information öffentlich verfügbar und es ist daher wichtig, dass eine Ausbesserung so schnell wie möglich vorhanden ist. Im letzten Fall könnte keine öffentliche Information vorliegen. In diesem Fall gibt es ein paar Möglichkeiten, wie mit dem Problem umzugehen ist:

- Wenn es ein triviales Problem ist (wie unsichere temporäre Dateien), gibt es keine Notwendigkeit, das Problem geheim zu halten, und eine Ausbesserung sollte erstellt und veröffentlicht werden.
- Wenn es sich um ein ernst zunehmendes Problem handelt (aus der Ferne ausnutzbar, Möglichkeit, Root-Rechte zu bekommen), ist es vorzugwürdig, die Information mit anderen Linux-Anbietern zu teilen und eine Veröffentlichung zu koordinieren. Das Sicherheitsteam hat Kontakte zu verschiedenen Organisationen und Individuen und kann das erledigen.

Wenn die Person, die das Problem gemeldet hat, darum bittet, die Informationen nicht bekanntzugeben, sollte das respektiert werden, mit der offensichtlichen Ausnahme der Mitteilung an das Sicherheitsteam (der Entwickler sollte sichergehen, dass er dem Sicherheitsteam mitteilt, dass die Informationen nicht bekanntgegeben werden sollen).

Beachten Sie, dass in Fällen der Geheimhaltung der Entwickler auch keine Ausbesserung nach Unstable (oder sonst irgendwo hin) hochladen darf, da die Änderungsübersicht für Unstable öffentlich zugänglich ist.

Es gibt zwei Gründe, um Informationen zu veröffentlichen, selbst wenn um Geheimhalten gebeten wurde oder diese notwendig ist: Das Problem ist schon zu lange bekannt oder es wurde öffentlich bekannt.

Erstellen eines Pakets

Wenn ein neues Paket erstellt wird, das ein Sicherheitsproblem löst, ist die wichtigste Richtlinie, so wenige Änderungen wie möglich vorzunehmen. Menschen hängen von demselben Verhalten einer Veröffentlichung ab. Jede Veränderung könnte also das System von jemanden unbenutzbar machen. Dies gilt besonders für Bibliotheken: Der Entwickler muss sichergehen, dass er niemals die API oder ABI verändert, egal wie klein die Änderungen sind.

Das bedeutet, dass das Verwenden einer neuen Version des Originalprogramms keine gute Lösung ist. Stattdessen sollten die relevanten Veränderung zurückportiert werden. Gewöhnlich werden die Programmautoren dabei gegebenenfalls behilflich sein, wenn Debians Sicherheitsteam nicht helfen kann.

In einigen Fällen ist es nicht möglich, Sicherheitsverbesserungen zurückzuportieren, z.B. wenn große Mengen des Quellcodes verändert oder neu geschrieben werden müssten. Wenn das eintritt, kann es notwendig werden, eine neue Version des Originalprogramms zu verwenden. Das sollte aber immer im Voraus mit dem Sicherheitsteam abgestimmt werden.

Damit hängt ein anderer wichtiger Punkt zusammen: Entwickler müssen immer ihre Änderungen testen. Wenn es einen Exploit gibt, sollte der Entwickler versuchen, ob er tatsächlich mit dem ungepatchten Paket gelingt und im ausgebesserten Paket scheitert. Der Entwickler sollte auch den gewöhnlichen Gebrauch ausprobieren, da manchmal eine Sicherheitsausbesserung fast unmerklich den normalen Gebrauch beeinträchtigt.

Zu guter Letzt ein paar technische Dinge, die Entwickler bedenken sollten:

- Stellen Sie sicher, dass Sie sich in Ihrer Debian-Änderungsübersicht auf die richtige Distribution beziehen. Für Stable ist das stable-security und für Testing testing-security. Beziehen Sie sich nicht auf <codename>-proposed-updates.
- Stellen Sie sicher, dass die Versionsnummer korrekt ist. Sie muss größer als die des aktuellen Pakets sein, aber niedriger als die Paketversionen in späteren Distributionen. Für Testing bedeutet das, dass es eine höhere Version als in Unstable sein muss. Falls es dort keine gibt (Testing und Unstable haben z.B. die gleichen Versionen), laden Sie zuerst die neue Version nach Unstable hoch.
- Laden Sie nicht nur die Quellen hoch (source-only upload), wenn Ihr Paket nur binäre Pakete enthält (binary-all). Die Buildd-Infrastruktur wird diese nicht erstellen.
- Wenn Sie ein Paket kompilieren, stellen Sie sicher, dass Sie es auf einem reinen System kompilieren, auf dem nur Pakete aus der Distribution installiert sind, für die Sie das Paket erstellen. Wenn Sie selbst ein solches System nicht haben, können Sie es mit einer Maschine von debian.org versuchen (siehe http://db.debian.org/machines.cgi) oder setzen Sie chroot ein (die Pakete `pbuilder` und `debootstrap` können dafür hilfreich sein).

Hochladen von Sicherheitsausbesserungen

Nachdem der Entwickler ein neues Paket erstellt und getestet hat, muss es hochgeladen werden, damit es in den Archiven installiert werden kann. Sicherheitsrelevante Dateien werden nach ftp://security.debian.org/pub/SecurityUploadQueue/ hochgeladen.

Wenn eine in die Sicherheitswarteschleife hochgeladene Datei akzeptiert wurde, wird das Paket automatisch für alle Architekturen neu erstellt und zur Überprüfung durch das Sicherheitsteam abgelegt.

Nur das Sicherheitsteam kann auf hochgeladene Dateien, die auf Annahme oder Überprüfung warten, zugreifen. Das ist notwendig, da es Ausbesserungen für Sicherheitsprobleme geben könnte, die noch nicht offengelegt werden dürfen.

Wenn ein Mitglied des Sicherheitsteams ein Paket akzeptiert, wird es auf security.debian.org und als passendes <codename>-proposed-updates auf ftp-master oder im Non-US-Archiv installiert.

Die Sicherheitsankündigung

Sicherheitsankündigungen werden vom Sicherheitsteam geschrieben und veröffentlicht. Allerdings macht es ihnen gewiss nichts aus, wenn ein Paketbetreuer den Text (teilweise) für sie erstellt. Informationen, die in einer Anweisung enthalten seien sollten, werden in 'Debian-Sicherheits-Ankündigungen' auf Seite 138 beschrieben.

7.4 Paketsignierung in Debian

Dieser Abschnitt könnte auch mit "Wie man sein Debian GNU/Linux-System sicher upgraded/aktualisiert" überschrieben werden. Es verdient hauptsächlich deshalb einen eigenen Abschnitt, weil es einen wichtigen Teil der Infrastruktur der Sicherheit darstellt. Die Signierung von Paketen ist ein wichtiges Thema, da es die Manipulation von Paketen in Spiegel und von heruntergeladenen Dateien durch Man-in-the-Middle-Angriffen verhindert. Die automatische Aktualisierung von Software ist eine wichtige Fähigkeit, aber es ist auch wichtig, Gefahren für die Sicherheit zu entfernen, die die Verbreitung von Trojanern und den Einbruch ins System während der Aktualisierung fördern können.[3]

Derzeit (Stand Februar 2006) stellt Debian keine signierten Pakete für die Distribution zur Verfügung, und die *Woody*- und *Sarge*-Veröffentlichung (3.0 und 3.1) werden diese Fähigkeit nicht integrieren. Es gibt eine Lösung für signierte Pakete, die – hoffentlich – in der nächsten Veröffentlichung (Codename *Etch*) enthalten sein wird. Diese neue Fähigkeit ist in apt 0.6 enthalten (ist im Moment Bestandteil der *Unstable*-Distribution, vergleiche 'Secure Apt' auf der nächsten Seite).

Dieses Problem wird besser im Strong-Distribution-Howto (`http://www.cryptnet.net/fdp/crypto/strong_distro.html`) von V. Alex Brennen beschrieben.

7.4.1 Der vorgeschlagene Plan zur Prüfung von Paketsignaturen

Der derzeitige Plan zur Prüfung von Paketsignaturen mit `apt` ist:

- Die `Release`-Datei enthält die MD5-Summe von `Packages.gz` (die die MD5-Summen der Pakete enthält) und wird signiert. Die Signatur stammt aus einer vertrauenswürdigen Quelle.
- Diese signierte `Release`-Datei wird beim "apt-get update" herunter geladen und zusammen mit `Packages.gz` gespeichert.
- Wenn ein Paket installiert werden soll, wird es zuerst herunter geladen, und dann wird die MD5-Summe erstellt.

[3] Einige Betriebssystem wurden schon von Problemen mit automatischen Aktualisierungen heimgesucht, wie z.B. die Mac OS X Software-Update-Verwundbarkeit (`http://www.cunap.com/~hardingr/projects/osx/exploit.html`). FIXME: probably the Internet Explorer vulnerability handling certificate chains has an impact on security updates on Microsoft Windows.

- Die signierte `Release`-Datei wird überprüft (ob die Signatur in Ordnung ist) und die MD5-Summe der `Packages.gz`-Datei extrahiert. Die MD5-Summe der `Packages.gz`-Datei wird erstellt und geprüft, und - wenn sie übereinstimmt - wird die MD5-Summe des heruntergeladenen Paketes aus ihr extrahiert.
- Wenn die MD5-Summe des heruntergeladenen Paketes die gleiche ist wie in der `Packages.gz`-Datei, wird das Paket installiert. Andernfalls wird der Administrator alarmiert, und das Paket wird im Zwischenspeicher gehalten (so dass der Administrator entscheiden kann, ob es installiert werden soll oder nicht). Wenn das Paket nicht in `Packages.gz` enthalten ist, und der Administrator das System so konfiguriert hat, dass nur geprüfte Pakete installiert werden können, wird das Paket ebenfalls nicht installiert.

Durch diese Kette von MD5-Summen ist `apt` in der Lage, zu verifizieren, dass ein Paket aus einer bestimmten Veröffentlichung stammt. Dies ist zwar unflexibler als jedes Paket einzeln zu signieren, kann aber auch mit den unten aufgeführten Plänen kombiniert werden.

Im Moment ist diese Vorgehensweise vollständig in apt 0.6 enthalten (http://lists.debian.org/debian-devel/2003/debian-devel-200312/msg01986.html), das nun im Testing- und Unstable-Zweig verfügbar ist; weitere Informationen finden Sie unter 'Secure Apt' auf dieser Seite. Pakete, die ein Frontend für apt anbieten, müssen verändert werden, um an diese neue Fähigkeit angepasst zu werden. Das gilt für `aptitude`, das verändert (http://lists.debian.org/debian-devel/2005/03/msg02641.html) wurde, um zu dieser Vorgehensweise zu passen. Frontends, die bekanntermaßen zurzeit mit dieser Fähigkeit umgehen können, sind `aptitude` und `synaptic`.

Die Signierung von Paketen wurde innerhalb des Debian-Projekts ausführlich diskutiert. Mehr Informationen hierzu finden Sie unter http://www.debian.org/News/weekly/2001/8/ und http://www.debian.org/News/weekly/2000/11/.

7.4.2 Secure Apt

Die Veröffentlichung von apt 0.6, die in *Testing* (im Moment *Etch*) und in Unstable verfügbar ist, enthält *apt-secure* (auch als *Secure Apt* bekannt), das ein Werkzeug ist, mit dem ein Systemadministrator die Integrität von heruntergeladenen Paketen mit dem oben dargestellten Verfahren überprüfen kann. Diese Veröffentlichung enthält das Werkzeug `apt-key`, um neue Schlüssel zum Schlüsselbund von apt hinzuzufügen, welcher standardmäßig nur den aktuellen Signierungsschlüssel des Debian-Archivs enthält.

Diese Veränderungen basieren auf dem Patch für `apt` (verfügbar in Fehler #203741 (http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=203741)), der diese Erweiterung zur Verfügung stellt.

Secure Apt überprüft die Distribution mit der `Release`-Datei. Dies wurde schon unter 'Überprüfung der Distribution mit der `Release`-Datei' auf der nächsten Seite dargestellt. Typischerweise erfordert dieser Vorgang kein Mitwirken des Administrators. Aber jedes Jahr müssen Sie eingreifen[4], um den neuen Schlüssel des Archivs hinzuzufügen, wenn dieser ausgewech-

[4] Bis ein automatischer Mechanismus entwickelt wurde.

selt wurde. Weitere Informationen zu den dazu notwendigen Schritten finden Sie unter 'Auf sichere Weise einen Schlüssel hinzufügen' auf Seite 151.

Diese Fähigkeit befindet sich noch im Entwicklungsstadium. Wenn Sie glauben, dass Sie Fehler gefunden haben, stellen Sie zuerst sicher, dass Sie die neuste Version verwenden (da dieses Paket vor seiner endgültigen Veröffentlichung noch ziemlich verändern werden kann). Falls Sie die aktuelle Version benutzen, schicken Sie einen Fehlerbericht für das Paket `apt`.

7.4.3 Überprüfung der Distribution mit der `Release`-Datei

Dieser Abschnitt beschreibt, wie die Überprüfung der Distribution mit Hilfe der `Release`-Datei funktioniert. Dies wurde von Joey Hess geschrieben und ist auch im Debian-Wiki (`http://wiki.debian.org/SecureApt`) abrufbar.

Grundlegende Konzepte

Es gibt ein paar grundlegende Konzepte, die Sie brauchen, um den Rest dieses Abschnitts verstehen zu können.

Eine Prüfsumme ist eine Methode, bei der eine Datei auf eine relativ kurze Zahl heruntergekocht wird, mit der der Inhalt der Datei eindeutig identifiziert werden kann. Dies ist wesentlich schwieriger, als es zunächst erscheinen mag. Der am weitesten verbreitetste Typ von Prüfsummen, MD5, ist gerade dabei, unbrauchbar zu werden.

Verschlüsselung mit öffentlichen Schlüsseln fußt auf einem Schlüsselpaar: einem öffentlichen Schlüssel und einem privaten Schlüssel. Der öffentliche Schlüssel wird an die Allgemeinheit verteilt. Der private muss ein Geheimnis bleiben. Jeder der den öffentlichen Schlüssel hat, kann eine Nachricht verschlüsseln, so dass sie nur noch der Besitzer des privaten Schlüssels lesen kann. Es besteht deneben die Möglichkeit, mit einem privaten Schlüssel eine Datei zu signieren. Wenn eine Datei mit einer digitalen Unterschrift versehen wurde, kann jeder, der den öffentlichen Schlüssel hat, überprüfen, ob die Datei mit diesem Schlüssel unterschrieben wurde. Ohne den privaten Schlüssel lässt sich eine solche Signatur nicht nachmachen.

Diese Schlüssel bestehen aus ziemlich langen Zahlen (1024 oder 2048 Ziffern oder sogar länger). Damit sie leichter zu verwenden sind, haben sie eine kürzere Schlüssel-ID (eine Zahl mit nur acht oder 16 Stellen), mit der sie bezeichnet werden können.

Secure Apt verwendet `gpg`, um Dateien zu unterschreiben und ihre Unterschriften zu überprüfen.

Mit dem Programm `apt-key` wird der Schlüsselbund von GPG für Secure Apt verwaltet. Der Schlüsselbund befindet sich in der Datei `/etc/apt/trusted.gpg` (nicht zu verwechseln mit der verwandten, aber nicht sehr interessanten Datei `/etc/apt/trustdb.gpg`). `apt-key` kann dazu verwendet werden, die Schlüssel im Schlüsselbund anzuzeigen oder um Schlüssel hinzuzufügen oder zu entfernen.

Prüfsummen der `Release`-Datei

Jedes Archiv von Debian enthält eine `Release`-Datei, die jedesmal aktualisiert wird, wenn ein Paket im Archiv geändert wird. Unter anderem enthält die `Release`-Datei MD5-Summen von anderen Dateien, die sich im Archiv befinden. Ein Auszug einer `Release`-Datei:

```
MD5Sum:
 6b05b392f792ba5a436d590c129de21f            3453 Packages
 1356479a23edda7a69f24eb8d6f4a14b            1131 Packages.gz
 2a5167881adc9ad1a8864f281b1eb959            1715 Sources
 88de3533bf6e054d1799f8e49b6aed8b             658 Sources.gz
```

Die `Release`-Datei enthält auch SHA1-Prüfsummen, was nützlich ist, wenn MD5-Summen vollständig unbrauchbar sind. Allerdings unterstützt apt SHA1 noch nicht.

Werfen wir einen Blick in eine `Paket`datei: wir sehen weitere MD5-Summen, eine für jedes darin aufgeführte Paket. Beispiel:

```
Package: uqm
Priority: optional
...
Filename: unstable/uqm_0.4.0-1_i386.deb
Size: 580558
MD5sum: 864ec6157c1eea88acfef44d0f34d219
```

Mit diesen beiden Prüfsummen kann überprüft werden, ob Sie eine getreue Kopie der `Paket`datei, also mit einer MD5-Summe, die mit der in der `Release`-Datei übereinstimmt, heruntergeladen haben. Und wenn ein einzelnes Paket heruntergeladen wird, kann auch die MD5-Summe mit dem Inhalt der `Paket`datei verglichen werden. Wenn bei einem dieser Schritte ein Fehler auftauchen sollte, bricht Apt den Vorgang ab.

Nichts davon ist neu in Secure Apt, aber es bietet die Grundlage dafür. Beachten Sie, dass es bis jetzt eine Datei gibt, die Apt nicht überprüfen kann: die `Release`-Datei. Bei Secure Apt dreht sich alles darum, dass Apt die `Release`-Datei überprüft, bevor es irgendetwas anderes damit macht. Wenn man das schafft, besteht eine lückenlose Authentifizierungskette von dem Paket, das Sie installieren möchten, bis zum Anbieter des Pakets.

Überprüfung der `Release`-Datei

Damit die `Release`-Datei überprüft werden kann, wird sie mit GPG signiert. Diese Unterschrift kommt in die Datei `Release.gpg`, die mit der `Release`-Datei abgerufen werden kann. Sie sieht in etwa so[5] aus, obwohl sich für gewöhnlich nur GPG ihren Inhalt ansieht:

[5] Genau genommen handelt es sich um eine ASCII-armored abgetrennte GPG-Signatur.

```
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.1 (GNU/Linux)

iD8DBQBCqKO1nukh8wJbxY8RAsfHAJ9hu8oGNRAl2MSmP5+z2RZb6FJ8kACfWvEx
UBGPVc7jbHHsg78EhMBlV/U=
=x6og
-----END PGP SIGNATURE-----
```

`Release.gpg` mit `Apt` überprüfen

Wenn Secure Apt eine `Release`-Datei herunterlädt, lädt es auch immer die `Release.gpg`-Datei herunter. Falls dies misslingen sollte oder die Signatur nicht stimmt, wird es eine Rückmeldung machen und hinweisen, dass die Paketdateien, auf die die `Release`-Datei verweist, und alle darin enthaltenen Pakete von einer nicht vertrauenswürdigen Quelle stammen. So würde dies während `apt-get update` aussehen:

```
W: GPG error: http://ftp.us.debian.org testing Release: The following signatu
 couldn't be verified because the public key is not available: NO_PUBKEY 0109
```

Beachten Sie, dass die zweite Hälfte der langen Nummer die Schlüssel-ID des Schlüssels ist, von dem Apt nichts weiß. Im Beispiel ist sie 2D230C5F.

Falls Sie diese Warnung ignorieren und später versuchen, ein Paket zu installieren, wird Sie Apt nochmals warnen:

```
WARNUNG: Die folgenden Pakete können nicht authentifiziert werden!
  libglib-perl libgtk2-perl
Diese Pakete ohne Überprüfung installieren [j/N]?
```

Wenn Sie nun J drücken, haben Sie keine Möglichkeit festzustellen, ob die Datei, die Sie bekommen, wirklich diejenige ist, die Sie auch installieren möchten, oder ob sie eine ganz andere ist, die Ihnen jemand, der die Verbindung mit dem Server abgefangen hat[6] mit einer gemeinen Überraschung, unterschieben will.

Sie können diese Abfragen abschalten, indem Sie `apt` mit `--allow-unauthenticated` laufen lassen.

Es lohnt sich auch noch darauf hinzuweisen, dass der Installer von Debian während des Debootstraps des Basissystems, solange Apt noch nicht verfügbar ist, denselben Mechanismus mit signierten `Release`-Dateien verwendet. Der Installer benutzt sogar dieses Verfahren, um Teile von sich selbst zu überprüfen, die er aus dem Netz gezogen hat. Debian signiert im Moment nicht die `Release`-Dateien auf den CDs. Apt kann aber so eingerichtet werden, dass es immer den Paketen von CDs vertraut, so dass dies nicht ein so großes Problem darstellt.

[6] Oder Ihren DNS vergiftet hat oder den Server spooft oder die Datei auf einem Spiegel platziert hat, den Sie verwenden, oder …

Wie man Apt sagt, wem es vertrauen soll

Die ganze Sicherheit des Verfahrens beruht also darauf, dass es eine `Release.gpg`-Datei gibt, die eine `Release`-Datei signiert, und dass diese Signatur von `apt` mit Hilfe von GPG überprüft wird. Dazu muss es den öffentlichen Schlüssel der Person kennen, die die Datei unterschrieben hat. Diese Schlüssel werden in Apts eigenem Schlüsselbund (`/etc/apt/trusted.gpg`) gespeichert. Bei der Verwaltung dieser Schlüssel kommt Secure Apt ins Spiel.

Standardmäßig befindet sich bei Debian-Systemen der Schlüssel des Debian-Archivs im Schlüsselbund.

```
# apt-key list
/etc/apt/trusted.gpg
--------------------
pub   1024D/4F368D5D 2005-01-31 [expires: 2006-01-31]
uid                  Debian Archive Automatic Signing Key (2005) >ftpmaster&d
an.org<
```

Im Beispiel ist 4F368D5D die Schlüssel-ID. Beachten Sie, dass dieser Schlüssel nur für ein Jahr gültig ist. Debian tauscht die Schlüssel als letzte Verteidigungslinie gegen Sicherheitsrisiken, die das Knacken eines Schlüssels umfassen, regelmäßig aus.

Mit dem Schlüssel des Archivs wird Apt dem offiziellen Archiv von Debian vertrauen. Wenn Sie aber weitere Paketdepots zu `/etc/apt/sources.list` hinzufügen wollen, müssen Sie Apt Ihre Schlüssel mitteilen, wenn Sie wollen, dass Apt ihnen vertraut. Sobald Sie den Schlüssel haben und ihn überprüft haben, müssen Sie nur `apt-key add` *`Datei`* laufen lassen, um den Schlüssel hinzuzufügen. Der schwierigste Teil dabei ist, den Schlüssel zu bekommen und ihn zu überprüfen.

Den Schlüssel für Paketdepots finden

Es gibt noch keinen standardisierten Ort, wo sich der Schlüssel für ein Paketdepot befindet. Es besteht die grobe Übereinkunft, dass der Schlüssel auf der Webseite des Paketdepots oder im Depot selbst zu finden sein sollte. Wie gesagt ist dies kein echter Standard, so dass Sie ihn unter Umständen suchen müssen.

Der Schlüssel des Debian-Archivs ist unter `http://ftp-master.debian.org/ziyi_key_2006.asc` (ersetzen Sie 2006 mit dem aktuellen Jahr) erhältlich.[7]

`gpg` besitzt mit den Schlüsselservern eine standardisierte Möglichkeit, Schlüssel zu verbreiten. Damit kann GPG einen Schlüssel herunterladen und ihn zum Schlüsselbund hinzufügen. Beispiel:

```
$ gpg --keyserver pgpkeys.mit.edu --recv-key 2D230C5F
```

[7] "Ziyi" ist offensichtlich der Name einer chinesischen Schauspielerin (`http://de.wikipedia.org/wiki/Ziyi_Zhang`).

```
gpg: requesting key 2D230C5F from hkp server pgpkeys.mit.edu
gpg: key 2D230C5F: public key "Debian Archive Automatic Signing Key (2006) <f
aster@debian.org>" imported
gpg: Anzahl insgesamt bearbeiteter Schlüssel: 1
gpg:                              importiert: 1
```

Sie können dann den Schlüssel aus Ihrem Schlüsselbund exportieren und ihn an `apt-key` weiterreichen:

```
$ gpg -a --export 2D230C5F | sudo apt-key add -
gpg: kein uneingeschränkt vertrauenswürdiger Schlüssel 080F67F4 gefunden
OK
```

Die Warnung »gpg: kein uneingeschränkt vertrauenswürdiger Schlüssel 080F67F4 gefunden« bedeutet, dass GPG nicht so konfiguriert wurde, um einem Schlüssel vollständig zu vertrauen. Das Zuweisen von Vertrauensstufen ist Teil des Web-of-Trust von OpenPGP, was hier nicht Gegenstand ist. Daher ist die Warnung unproblematisch. Für gewöhnlich wird dem eignen Schlüssel eines Benutzers vollständig vertraut.

Auf sichere Weise einen Schlüssel hinzufügen

Indem Sie einen Schlüssel zu Apts Schlüsselbund hinzufügen, lassen Sie Apt wissen, dass es allem vertrauen soll, was mit diesem Schlüssel signiert wurde. Dadurch stellen Sie sicher, dass Apt nichts installiert, was nicht vom Inhaber des privaten Schlüssels signiert wurde. Mit ausreichender Paranoia erkennen Sie aber, dass dies das Problem nur um eine Stufe verlagert: Anstatt sich nun darum Sorgen zu machen, ob ein Paket oder eine `Release`-Datei korrekt ist, müssen Sie überprüfen, ob Sie tatsächlich den richtigen Schlüssel haben. Ist die Datei `http://ftp-master.debian.org/ziyi_key_2006.asc`, die oben erwähnt wird, wirklich der Signierungsschlüssel des Debian-Archivs oder wurde sie verändert (oder wird gar in diesem Dokument gelogen)?

Es ist gut, in Sicherheitsfragen Vorsicht walten zu lassen. Aber ab hier wird es schwieriger, Dinge zu überprüfen. `gpg` arbeitet mit dem Konzept der Kette des Vertrauens (chain of trust), die bei jemandem beginnt, dem Sie vertrauen und der einen anderen Schlüssel unterschreibt usw., bis Sie beim Schlüssel des Archivs sind. Wenn Sie vorsichtig sind, wollen Sie nachprüfen, dass Ihr Archivschlüssel von einem Schlüssel unterschrieben wurde, dem Sie vertrauen können, weil seine Kette des Vertrauens zu jemandem zurückgeht, den Sie persönlich kennen. Dazu sollten Sie eine Debian-Konferenz oder eine lokale LUG zum Unterschreiben der Schlüssel besuchen.[8]

Wenn Sie diese Sicherheitsbedenken nicht teilen (können), unternehmen Sie was auch immer Sie passend finden, wenn Sie eine neue Apt-Quelle oder einen neuen Schlüssel verwenden.

[8] Nicht alle Schlüssel der Apt-Depots sind überhaupt mit einem anderen Schlüssel unterschrieben. Vielleicht hat derjenige, der das Depot einrichtet, keinen anderen Schlüssel zur Verfügung, oder vielleicht ist es ihm unangenehm, einen Schlüssel mit einer derartig wichtigen Funktion mit seinem Hauptschlüssel zu unterschreiben. Hinweise, wie man einen Schlüssel für ein Depot einrichtet, finden Sie unter 'Prüfung von Debian-fremden Quellen' auf Seite 161.

Sie könnten demjenigen, der den Schlüssel anbietet, eine Mail schreiben, um den Schlüssel zu überprüfen. Oder Sie vertrauen auf Ihr Glück und gehen davon aus, dass Sie den richten heruntergeladen haben. Das wichtige ist, dass Secure Apt, indem es das Problem darauf reduziert, welchen Archivschlüsseln Sie vertrauen, Sie so vorsichtig und sicher vorgehen lässt, wie es Ihnen passend und notwendig erscheint.

Die Integrität eines Schlüssels überprüfen

Sie können dazu sowohl den Fingerabdruck als auch die Unterschriften des Schlüssels überprüfen. Den Fingerabdruck kann man aus verschiedenen Quellen erhalten. Sie können im Buch The Debian System (http://debiansystem.info/readers/changes/547-ziyi-key-2006) nachsehen, im IRC mit Debian-Entwicklern reden oder Mailinglisten lesen, wo ein Wechsel des Schlüssel angekündigt werden wird, oder jede andere erdenkliche Methode verwenden, um den Fingerabdruck zu überprüfen. Zum Beispiel können Sie auch Folgendes machen:

```
$ GET http://ftp-master.debian.org/ziyi_key_2006.asc | gpg --import
gpg: key 2D230C5F: public key "Debian Archive Automatic Signing Key (2006)
  <ftpmaster&debian.org>" imported
gpg: Anzahl insgesamt bearbeiteter Schlüssel: 1
gpg:                              unverändert: 1
$ gpg --check-sigs --fingerprint 2D230C5F
pub   1024D/2D230C5F 2006-01-03 [expires: 2007-02-07]
      Key fingerprint = 0847 50FC 01A6 D388 A643  D869 0109 0831 2D23 0C5F
uid   Debian Archive Automatic Signing Key (2006) <ftpmaster@debian.org>
sig!3        2D230C5F 2006-01-03  Debian Archive Automatic Signing Key
                                  (2006) <ftpmaster@debian.org>
sig!         2A4E3EAA 2006-01-03  Anthony Towns <aj@azure.humbug.org.au>
sig!         4F368D5D 2006-01-03  Debian Archive Automatic Signing Key
                                  (2005) <ftpmaster@debian.org>
sig!         29982E5A 2006-01-04  Steve Langasek <vorlon@dodds.net>
sig!         FD6645AB 2006-01-04  Ryan Murray <rmurray@cyberhqz.com>
sig!         AB2A91F5 2006-01-04  James Troup <james@nocrew.org>
```

und dann von Ihrem Schlüssel (oder einem Schlüssel, dem Sie vertrauen) den Pfad des Vertrauens (http://www.de.debian.org/doc/manuals/securing-debian-howto/ch7#s-deb-pack-sign) zu wenigstens einem der Schlüssel, der verwendet wurde, um den Archivschlüssel zu unterschreiben, überprüfen. Wenn Sie vorsichtig sein wollen, sollten Sie Apt mitteilen, dass es dem Schlüssel nur vertrauen darf, wenn es einen passenden Pfad gefunden hat:

```
$ gpg --export -a 2D230C5F | sudo apt-key add -
Ok
```

Der aktuelle Schlüssel ist mit dem vorhergehenden Archivschlüssel unterschrieben, so dass Sie theoretisch auf Ihrem alten Vertrauen aufbauen können.

Der jährliche Austausch des Archivschlüssels von Debian

Wie schon oben erwähnt wird der Schlüssel, mit dem das Debian-Archiv signiert wird, jedes Jahr im Januar ausgetauscht. Da Secure Apt noch jung ist, haben wir noch nicht sehr viel Erfahrung damit und es gibt noch ein paar haarige Stellen.

Im Januar 2006 wurde ein neuer Schlüssel für 2006 erstellt und die `Release`-Datei wurde damit unterschrieben. Um aber zu vermeiden, dass Systeme, die noch den alten Schlüssel von 2005 verwenden, nicht mehr korrekt arbeiten, wurde die `Release`-Datei auch mit dem alten Schlüssel unterschrieben. Es war geplant, dass Apt je nach dem verfügbaren Schlüssel eine der beiden Unterschriften akzeptieren würde. Aber es zeigte sich ein Fehler in Apt, da es sich weigerte, der Datei zu vertrauen, wenn es nicht beide Schlüssel hatte und somit beide Unterschriften überprüfen konnte. Dies wurde in der Version 0.6.43.1 ausgebessert. Es gab auch Verwirrung darüber, wie der Schlüssel an Benutzer verteilt wird, die bereits Secure Apt auf ihrem System laufen lassen. Am Anfang wurde er auf die Webseite hochgeladen, ohne Ankündigung und ohne eine echte Möglichkeit, ihn zu überprüfen, und die Benutzer mussten ihn per Hand herunterladen.

Aufgrund dieser Erfahrung werden die Dinge in 2007 wie folgt ablaufen:

- Anfang Januar wird ein neuer Schlüssel für 2007 erstellt. Vielleicht gibt es dieses Mal auch eine Ankündigung und eine gute Kette des Vertrauens.
- Die `Release`-Datei wird mit diesem Schlüssel unterschrieben werden, während sie auch mit dem Schlüssel von 2006 unterschrieben ist. Apt und andere Werkzeuge werden beide Unterschriften akzeptieren.
- Zuvor wird ein neues Paket, `debian-server-keyring`, auf jedem System installiert werden. Es wird aktualisiert, damit es den Schlüssel von 2007 enthält. Wenn Benutzer zur neuen Version wechseln, wird mit `apt-key` ihr Schlüsselbund aktualisiert werden. Der Schlüssel von 2006 wird entfernt und der von 2007 hinzugefügt.
- Der Schlüssel von 2006 verliert am 31. Januar 2007 seine Gültigkeit.

Noch nicht ganz sicher ist, was mit denen passiert, die im Januar gar keine Aktualisierung durchführen, und wie diese Aktualisierung für Personen abläuft, die Stable laufen lassen, wenn Secure Apt auch dort verfügbar ist.

Bekannte Probleme bei der Prüfung

Ein nicht offensichtliches Problem ist, dass Secure Apt nicht funktioniert, wenn Ihre Uhr sehr verstellt ist. Wenn sie auf ein Datum in der Vergangenheit wie 1999 eingestellt ist, wird Apt mit einer nichts sagenden Ausgabe wie dieser abbrechen:

```
W: GPG error: http://archive.progeny.com sid Release: Unknown error executing
```

Dagegen macht `apt-key` das Problem deutlich:

```
gpg: key 2D230C5F was created 192324901 seconds in the future (time warp or c
gpg: key 2D230C5F was created 192324901 seconds in the future (time warp or c
pub   1024D/2D230C5F 2006-01-03
uid                  Debian Archive Automatic Signing Key (2006) <ftpmaster@d
```

Falls die Uhr nicht zu weit vorgeht, behandelt Apt die Schlüssel als abgelaufen.

Wenn Sie Testing oder Unstable verwenden, gibt es ein Problem, wenn Sie in letzter Zeit nicht `apt-get update` ausgeführt haben und mit `apt-get` ein Paket installieren möchten. Apt könnte sich darüber beschweren, dass es nicht authentifiziert werden konnte (Warum passiert das bloß?). `apt-get update` löst das Problem.

Prüfung von Hand

Für den Fall, dass Sie nun zusätzliche Sicherheitsprüfungen einführen wollen, aber nicht die neuste Version von apt einsetzen wollen oder können[9], können Sie das folgende Skript von Anthony Towns benutzen. Dieses Skript führt automatisch neue Sicherheitsüberprüfungen durch, damit ein Nutzer sicher gehen kann, dass die Software, die er herunterlädt, die gleiche ist wie die, die von Debian bereitgestellt wird. Das verhindert, dass sich Debian-Entwickler in ein fremdes System einhacken können, ohne dass eine Zurechnung und Rückverfolgung möglich wäre, die durch das Hochladen eines Pakets auf das Hauptarchiv gewährleistet werden. Es kann auch verhindern, dass ein Spiegel etwas fast genau abbildet, das aber eben doch nicht ganz wie in Debian, oder dass veraltete Versionen von instabilen Paketen mit bekannten Sicherheitslücken zur Verfügung gestellt werden.

Dieser Beispielscode, umbenannt nach `apt-check sigs`, sollte auf die folgende Art benutzt werden:

```
# apt-get update
# apt-check-sigs
(... Ergebnisse ...)
# apt-get dist-upgrade
```

Zuerst müssen Sie jedoch Folgendes tun:

- Holen Sie sich den Schlüssel, den die Archiv-Software verwendet, um `Release`-Dateien zu signieren, `http://ftp-master.debian.org/ziyi_key_2006.asc` und fügen Sie ihn `~/.gnupg/trustedkeys.gpg` hinzu (was standardmäßig von `gpgv` benutzt wird).

```
gpg --no-default-keyring --keyring trustedkeys.gpg --import ziyi_key_
```

[9] Entweder weil Sie Stable (*Sarge*) oder eine ältere Veröffentlichung verwenden, oder weil Sie nicht die neuste Version von Apt einsetzen wollen, obwohl wir das Testen wirklich schätzen würden.

- Entfernen Sie alle Zeilen aus `/etc/apt/sources.list`, die nicht die normale "dists"-Struktur benutzen, oder ändern Sie das Skript, so dass es auch mit denen funktioniert.
- Ignorieren Sie die Tatsache, dass Sicherheitsaktualisierungen von Debian keine signierten `Release`-Dateien haben, und das `Quell`dateien (noch) keine richtigen Prüfsummen in der `Release`-Datei haben.
- Bereiten Sie sich darauf vor, zu prüfen, dass die richtigen Quellen durch den richtigen Schlüssel signiert wurden.

Dies ist der Beispielscode für `apt-check-sigs`. Die neuste Fassung ist unter http://people.debian.org/~ajt/apt-check-sigs erhältlich. Dieser Code befindet sich im Moment noch im Beta-Stadium. Für weitere Informationen sollten Sie http://lists.debian.org/debian-devel/2002/debian-devel-200207/msg00421.html lesen.

```
#!/bin/bash

# Copyright (c) 2001 Anthony Towns <ajt@debian.org>
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

rm -rf /tmp/apt-release-check
mkdir /tmp/apt-release-check || exit 1
cd /tmp/apt-release-check

>OK
>MISSING
>NOCHECK
>BAD

arch=‘dpkg --print-installation-architecture‘

am_root () {
        [ ‘id -u‘ -eq 0 ]
}

get_md5sumsize () {
        cat "$1" | awk ’/^MD5Sum:/,/^SHA1:/’ |
```

```
          MYARG="$2" perl -ne '@f = split /\s+/; if ($f[3] eq $ENV{"MYARG"})
print "$f[1] $f[2]\n"; exit(0); }'
}

checkit () {
        local FILE="$1"
        local LOOKUP="$2"

        Y="`get_md5sumsize Release "$LOOKUP"`"
        Y="`echo "$Y" | sed 's/^ *//;s/  */ /g'`"

        if [ ! -e "/var/lib/apt/lists/$FILE" ]; then
                if [ "$Y" = "" ]; then
                        # No file, but not needed anyway
                        echo "OK"
                        return
                fi
                echo "$FILE" >>MISSING
                echo "MISSING $Y"
                return
        fi
        if [ "$Y" = "" ]; then
                echo "$FILE" >>NOCHECK
                echo "NOCHECK"
                return
        fi
        X="`md5sum < /var/lib/apt/lists/$FILE | cut -d\  -f1` `wc -c < /var/l
/apt/lists/$FILE`"
        X="`echo "$X" | sed 's/^ *//;s/  */ /g'`"
        if [ "$X" != "$Y" ]; then
                echo "$FILE" >>BAD
                echo "BAD"
                return
        fi
        echo "$FILE" >>OK
        echo "OK"
}

echo
echo "Checking sources in /etc/apt/sources.list:"
echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"
echo
(echo "You should take care to ensure that the distributions you're downloadi
"
echo "are the ones you think you are downloading, and that they are as up to"
echo "date as you would expect (testing and unstable should be no more than"
```

```
echo "two or three days out of date, stable-updates no more than a few weeks"
echo "or a month)."
) | fmt
echo

cat /etc/apt/sources.list |
  sed 's/^ *//' | grep '^[^#]' |
  while read ty url dist comps; do
        if [ "${url%%:*}" = "http" -o "${url%%:*}" = "ftp" ]; then
                baseurl="${url#*://}"
        else
                continue
        fi

        echo "Source: ${ty} ${url} ${dist} ${comps}"

        rm -f Release Release.gpg
        lynx -reload -dump "${url}/dists/${dist}/Release" >/dev/null 2>&1
        wget -q -O Release "${url}/dists/${dist}/Release"

        if ! grep -q '^' Release; then
                echo "  * NO TOP-LEVEL Release FILE"
                >Release
        else
                origline=`sed -n 's/^Origin: *//p' Release | head -1`
                lablline=`sed -n 's/^Label: *//p' Release | head -1`
                suitline=`sed -n 's/^Suite: *//p' Release | head -1`
                codeline=`sed -n 's/^Codename: *//p' Release | head -1`
                dateline=`grep "^Date:" Release | head -1`
                dscrline=`grep "^Description:" Release | head -1`
                echo "  o Origin: $origline/$lablline"
                echo "  o Suite: $suitline/$codeline"
                echo "  o $dateline"
                echo "  o $dscrline"

                if [ "${dist%%/*}" != "$suitline" -a "${dist%%/*}" != "$codel
                        echo "  * WARNING: asked for $dist, got $suitline/$co
                fi

                lynx -reload -dump "${url}/dists/${dist}/Release.gpg" >/dev/n
                wget -q -O Release.gpg "${url}/dists/${dist}/Release.gpg"

                gpgv --status-fd 3 Release.gpg Release 3>&1 >/dev/null 2>&1 |
                        if [ "$gpgcode" = "GOODSIG" ]; then
                            if [ "$err" != "" ]; then
                                echo "  * Signed by ${err# } key: ${rest#* }"
```

```
                            else
                                echo "  o Signed by: ${rest#* }"
                                okay=1
                            fi
                            err=""
                        elif [ "$gpgcode" = "BADSIG" ]; then
                            echo "  * BAD SIGNATURE BY: ${rest#* }"
                            err=""
                        elif [ "$gpgcode" = "ERRSIG" ]; then
                            echo "  * COULDN'T CHECK SIGNATURE BY KEYID: ${re
                            err=""
                        elif [ "$gpgcode" = "SIGREVOKED" ]; then
                            err="$err REVOKED"
                        elif [ "$gpgcode" = "SIGEXPIRED" ]; then
                            err="$err EXPIRED"
                        fi
                    done
                    if [ "$okay" != 1 ]; then
                        echo "  * NO VALID SIGNATURE"
                        >Release
                    fi)
        fi
        okaycomps=""
        for comp in $comps; do
                if [ "$ty" = "deb" ]; then
                        X=$(checkit "'echo "${baseurl}/dists/${dist}/${comp}/
                        Y=$(checkit "'echo "${baseurl}/dists/${dist}/${comp}/
                        if [ "$X $Y" = "OK OK" ]; then
                                okaycomps="$okaycomps $comp"
                        else
                                echo "  * PROBLEMS WITH $comp ($X, $Y)"
                        fi
                elif [ "$ty" = "deb-src" ]; then
                        X=$(checkit "'echo "${baseurl}/dists/${dist}/${comp}/
                        Y=$(checkit "'echo "${baseurl}/dists/${dist}/${comp}/
                        if [ "$X $Y" = "OK OK" ]; then
                                okaycomps="$okaycomps $comp"
                        else
                                echo "  * PROBLEMS WITH component $comp ($X,
                        fi
                fi
        done
        [ "$okaycomps" = "" ] || echo "  o Okay:$okaycomps"
        echo
done
```

```
echo "Results"
echo "~~~~~~~~"
echo

allokay=true

cd /tmp/apt-release-check
diff <(cat BAD MISSING NOCHECK OK | sort) <(cd /var/lib/apt/lists && find . -

cd /tmp/apt-release-check
if grep -q ^ UNVALIDATED; then
    allokay=false
    (echo "The following files in /var/lib/apt/lists have not been validated.
    echo "This could turn out to be a harmless indication that this script"
    echo "is buggy or out of date, or it could let trojaned packages get onto
    echo "your system."
    ) | fmt
    echo
    sed 's/^/    /' < UNVALIDATED
    echo
fi

if grep -q ^ BAD; then
    allokay=false
    (echo "The contents of the following files in /var/lib/apt/lists does not
    echo "match what was expected. This may mean these sources are out of dat
    echo "that the archive is having problems, or that someone is actively"
    echo "using your mirror to distribute trojans."
    if am_root; then
        echo "The files have been renamed to have the extension .FAILED and"
        echo "will be ignored by apt."
        cat BAD | while read a; do
            mv /var/lib/apt/lists/$a /var/lib/apt/lists/${a}.FAILED
        done
    fi) | fmt
    echo
    sed 's/^/    /' < BAD
    echo
fi

if grep -q ^ MISSING; then
    allokay=false
    (echo "The following files from /var/lib/apt/lists were missing. This"
    echo "may cause you to miss out on updates to some vulnerable packages."
    ) | fmt
    echo
```

```
    sed 's/^/    /' < MISSING
    echo
fi

if grep -q ^ NOCHECK; then
    allokay=false
    (echo "The contents of the following files in /var/lib/apt/lists could no
    echo "be validated due to the lack of a signed Release file, or the lack"
    echo "of an appropriate entry in a signed Release file. This probably"
    echo "means that the maintainers of these sources are slack, but may mean
    echo "these sources are being actively used to distribute trojans."
    if am_root; then
        echo "The files have been renamed to have the extension .FAILED and"
        echo "will be ignored by apt."
        cat NOCHECK | while read a; do
            mv /var/lib/apt/lists/$a /var/lib/apt/lists/${a}.FAILED
        done
    fi) | fmt
    echo
    sed 's/^/    /' < NOCHECK
    echo
fi

if $allokay; then
    echo 'Everything seems okay!'
    echo
fi

rm -rf /tmp/apt-release-check
```

Sie müssen vielleicht bei *Sid* diesen Patch verwenden, da `md5sum` ein '-' an die Summe anfügt, wenn die Ausgabe auf stdin erfolgt:

```
@@ -37,7 +37,7 @@
        local LOOKUP="$2"

        Y="`get_md5sumsize Release "$LOOKUP"`"
-       Y="`echo "$Y" | sed 's/^ *//;s/  */ /g'`"
+       Y="`echo "$Y" | sed 's/-//;s/^ *//;s/  */ /g'`"

        if [ ! -e "/var/lib/apt/lists/$FILE" ]; then
                if [ "$Y" = "" ]; then
@@ -55,7 +55,7 @@
                return
        fi
```

```
        X="‘md5sum < /var/lib/apt/lists/$FILE‘ ‘wc -c < /var/lib/apt/lists/$F
-       X="‘echo "$X" | sed ’s/^ *//;s/  */ /g’‘"
+       X="‘echo "$X" | sed ’s/-//;s/^ *//;s/  */ /g’‘"
        if [ "$X" != "$Y" ]; then
                echo "$FILE" >>BAD
                echo "BAD"
```

7.4.4 Prüfung von Debian-fremden Quellen

Beachten Sie, dass, wenn Sie die neuste Version von Apt (mit *Secure Apt*) einsetzen, kein zusätzlicher Aufwand auf Ihrer Seite notwendig sein sollte, wenn Sie keine Debian-fremden Quellen verwenden. In diesen Fällen erfordert `apt-get` eine zusätzliche Bestätigung. Dies wird verhindert, wenn `Release`- und `Release.gpg`-Dateien in den Debian-fremden Quellen zur Verfügung stehen. Die `Release`-Datei kann mit `apt-ftparchive` (ist in `apt-utils` 0.5.0 und später enthalten) erstellt werden, die `Release.gpg` ist nur die abgetrennte Signatur. Beide können mit folgender einfacher Prozedur erstellt werden:

```
$ rm -f dists/unstable/Release
$ apt-ftparchive release dists/unstable > dists/unstable/Release
$ gpg --sign -ba -o dists/unstable/Release.gpg dists/unstable/Release
```

7.4.5 Alternativer Entwurf zur Einzelsignierung von Paketen

Dieser zusätzliche Entwurf, jedes Paket einzeln zu signieren, erlaubt es, Pakete zu prüfen, selbst wenn sie nicht mehr in irgendeiner `Packages`-Datei erwähnt werden. Und so können auch Pakete von Dritten, für die es nie eine `Packages`-Datei gab, unter Debian installiert werden. Dies wird aber kein Standard werden.

Dieser Entwurf zur Paketsignierung kann mit `debsig-verify` und `debsigs` umgesetzt werden. Diese beiden Pakete können in einer .deb-Datei eingebettete Unterschriften erstellen und prüfen. Debian hat bereits jetzt die Möglichkeiten, dies zu tun. Aber es gibt keine Planung, dieses Regelwerk oder ähnliche Werkzeuge umzusetzen, da nunmehr das Schema mit der Signierung des Archivs bevorzugt wird. Die Werkzeuge werden dennoch für Benutzer und Administratoren von Archiven zur Verfügung gestellt, wenn sie diese Vorgehensweise bevorzugen.

Die aktuellen Versionen von `dpkg` (seit 1.9.21) beinhalten einen Patch (`http://lists.debian.org/debian-dpkg/2001/debian-dpkg-200103/msg00024.html`), der diese Funktionen zur Verfügung stellt, sobald `debsig-verify` installiert ist.

HINWEIS: Derzeit wird `/etc/dpkg/dpkg.cfg` standardmäßig mit der Option “no-debsig” ausgeliefert.

HINWEIS 2: Unterschriften von Entwicklern werden im Moment entfernt, wenn sie in das Paketarchiv gelangen, da die derzeit vorzugswürdige Methode die Überprüfung der `Release`-Datei ist, wie es oben beschrieben wurde.

Kapitel 8

Sicherheitswerkzeuge in Debian

FIXME: More content needed.

Debian stellt außerdem einige Sicherheitswerkzeuge zur Verfügung, die eine Debian-Maschine zum Zweck der Sicherheit passend einrichten können. Diese Zielsetzung schließt die Sicherung von Systeminformationen durch Firewalls (sowohl auf Paket- als auch auf Anwendungsebene), Eindringlingserkennung (netzwerk- und hostbasiert), Einschätzung der Verwundbarkeit, Antivirus, private Netzwerke und vieles mehr ein.

Seit Debian 3.0 (*woody* ist kryptographische Software in der Hauptdistribution integriert. OpenSSH und GNU Privacy Guard sind in der Standardinstallation enthalten. Außerdem befinden sich jetzt in Web-Browsern und Web-Servern, Datenbanken usw. starke Verschlüsselungsmechanismen. Eine weitergehende Eingliederung von Kryptographie ist für zukünftige Veröffentlichungen geplant. Aufgrund von Exportbeschränkungen in den USA wurde diese Software nicht mit der Hauptdistribution ausgeliefert, sondern war nur auf Seiten außerhalb der USA erhältlich.

8.1 Programme zur Fernprüfung der Verwundbarkeit

Die Werkzeuge, um Fernprüfungen der Verwundbarkeit durchzuführen, sind unter Debian: [1]

- `nessus`
- `raccess`
- `nikto` (Ersatz für `whisker`)

Das weitaus vollständigste und aktuellste Werkzeug ist `nessus`, welches aus einem Client (`nessus`) mit graphischer Benutzungsschnittstelle und einem Server (`nessusd`), der die programmierten Attacken startet, besteht. Nessus kennt verschiedene entfernten Verwundbarkeiten für einige Systeme, einschließlich Netzwerkanwendungen, FTP-Servern, WWW-Servern,

[1] Manche von ihnen sind erhältlich, wenn Sie das Paket `harden-remoteaudit` installieren.

usw. Die neusten Sicherheitsplugins sind sogar in der Lage, eine Web-Seite zu analysieren und zu versuchen, interaktive Inhalte zu entdecken, die zu einem Angriff genutzt werden können. Es existieren auch Java- und Win32-Clients, die benutzt werden können, um sich mit dem Nessus-Server zu verbinden. Diese sind jedoch in Debian nicht enthalten.

Beachten Sie, wenn Sie Woody einsetzen, dass das Nessus-Paket ziemlich veraltet ist (siehe Fehler #183524 (`http://bugs.debian.org/183524`)). Es ist nicht schwierig, die Pakete aus Unstable nach Woody zurückzuportieren. Falls es Ihnen dennoch Schwierigkeiten bereiten sollte, können Sie auch die zurückportierten Pakete von einem Mitentwickler verwenden. Sie sind unter `http://people.debian.org/~jfs/nessus/` verfügbar (diese Versionen können aber nicht so aktuelle sein wie die Versionen aus *Unstable*).

`nikto` ist ein Scanner zur Aufdeckung von Schwachstellen bei Webservern und kennt auch einige Anti-IDS-Taktiken (die meisten davon sind keine *Anti-IDS*-Taktiken mehr). Er ist einer der besten verfügbaren CGI-Scanner zur Erkennung von WWW-Servern und kann nur bestimmte Angriffe gegen ihn starten. Die Datenbank, die zum Scannen benutzt wird, kann sehr leicht geändert werden, um neue Informationen einzufügen.

8.2 Werkzeuge zum Scannen von Netzwerken

Debian bietet Ihnen einige Werkzeuge zum Scannen von Hosts (aber nicht zur Gefahrenabschätzung). Diese Programme werden in manchen Fällen von Scannern zur Gefahrenabschätzung zu einem ersten "Angriff" gegen entfernte Rechner genutzt, um festzustellen, welche Dienste angeboten werden. Unter Debian sind im Moment verfügbar:

- `nmap`
- `xprobe`
- `p0f`
- `knocker`
- `isic`
- `hping2`
- `icmpush`
- `nbtscan` (für die Prüfung von NetBIOS)
- `fragrouter`
- `strobe` (aus dem Paket `netdiag`)
- `irpas`

Während `xprobe` lediglich aus der Ferne das Betriebssystem erkennen kann (indem es TCP/IP-Fingerabdrücke benutzt, machen `nmap` und `knocker` beides: das Betriebssystem erkennen und die Ports eines entfernten Rechners scannen. Andererseits können `hping2` und `icmpush` für ICMP-Angriffstechniken benutzt werden.

`Nbtscan`, das speziell für SMB-Netzwerke entworfen wurde, kann benutzt werden, um IP-Netzwerke zu scannen und diverse Informationen von SMB-Servern zu ermitteln einschließlich der Nutzernamen, Netzwerknamen, MAC-Adressen, ...

Dagegen kann `fragrouter` dazu verwendet werden, um Systeme zur Eindringlingserkennung zu testen und um zu sehen, ob das NIDS mit fragmentierten Angriffen umgangen werden kann.

FIXME: Check Bug #153117 (`http://bugs.debian.org/153117`) (ITP fragrouter) to see if it's included.

FIXME: add information based on Debian Linux Laptop for Road Warriors (`http://www.giac.org/practical/gcux/Stephanie_Thomas_GCUX.pdf`) which describes how to use Debian and a laptop to scan for wireless (803.1) networks (link not there any more).

8.3 Interne Prüfungen

Derzeit kann lediglich das Programm `tiger` benutzt werden, um interne Prüfungen (auch "white box" genannt) eines Rechners vorzunehmen. Dabei wird festgestellt, ob das Dateisystem richtig aufgesetzt ist, welche Prozesse auf dem Rechner horchen, usw.

8.4 Testen des Quellcodes

Debian bietet einige Pakete an, die C/C++-Quellcode prüfen und Programmierfehler finden, die zu möglichen Sicherheitsmängeln führen können:

- `flawfinder`
- `rats`
- `splint`
- `pscan`

8.5 Virtual Private Networks (virtuelle private Netzwerke)

Ein virtuelles privates Netzwerk (VPN) ist eine Gruppe von zwei oder mehreren Computern, die typischerweise zu einem privaten Netzwerk mit begrenztem öffentlichen Netzwerkzugang verbunden sind und gesichert über ein öffentliches Netzwerk kommunizieren. VPNs können

einen einzelnen Rechner mit einem privaten Netzwerk verbinden (Client-Server) oder ein entferntes LAN mit einem privaten Netzwerk (Server-Server). VPNs verwenden Verschlüsselung, starke Authentifikation von entfernten Nutzern oder Hosts und Methoden, um Struktur des privaten Netzwerks zu verstecken.

Debian enthält einige Pakete, die zum Aufsetzen von verschlüsselten virtuellen privaten Netzwerken verwendet werden können:

- `vtun`
- `tunnelv` (Abschnitt non-US)
- `cipe-source`, `cipe-common`
- `tinc`
- `secvpn`
- `pptpd`
- `openvpn`
- `freeswan`, welches jetzt überholt ist. Sein Ersatz ist:
- `openswan` (`http://www.openswan.org/`)

FIXME: Update the information here since it was written with FreeSWAN in mind. Check Bug #237764 and Message-Id: <200412101215.04040.rmayr@debian.org>.

Das OpenSWAN-Paket ist wahrscheinlich die beste Wahl, da es nahezu mit allem zusammenarbeiten kann, das das IP-Security-Protokoll IPsec (RFC 2411) benutzt. Aber auch die anderen oben aufgeführten Pakete können Ihnen helfen, möglichst schnell einen sicheren Tunnel aufzusetzen. Das Point-to-Point-Tunneling-Protocol (PPTP) ist ein urheberrechtlich geschütztes Protokoll von Microsoft für VPN. Es wird unter Linux unterstützt, aber es sind einige schwere Sicherheitsprobleme bekannt.

Für weitere Informationen über IPsec und PPTP lesen Sie bitte das VPN-Masquerade-HOWTO (`http://www.tldp.org/HOWTO/VPN-Masquerade-HOWTO.html`), über PPP über SSH das VPN-HOWTO (`http://www.tldp.org/HOWTO/VPN-HOWTO.html`), das Cipe-Mini-HOWTO (`http://www.tldp.org/HOWTO/mini/Cipe+Masq.html`) und das PPP- und SSH-Mini-HOWTO (`http://www.tldp.org/HOWTO/mini/ppp-ssh/index.html`).

Es kann sich auch lohnen, sich Yavipin (`http://yavipin.sourceforge.net/`) anzusehen. Allerdings scheinen noch keine Pakete für Debian verfügbar zu sein.

8.5.1 Point-to-Point-Tunneling

Wenn Sie einen tunnelnden Server für eine gemischte Umgebung (sowohl Microsofts Betriebssystem als auch Linux-Clients) zur Verfügung stellen wollen und IPsec keine Möglichkeit ist

(da es nur in Windows 2000 und Windows XP enthalten ist), können Sie *PoPToP* (Point to Point Tunneling Server) verwenden. Er wird vom Paket `pptpd` bereitgestellt.

Wenn Sie Microsofts Authentifikation und Verschlüsselung mit dem Server verwenden wollen, die im Paket `ppp` enthalten sind, sollten Sie Folgendes aus der FAQ beachten:

```
Sie müssen nur dann PPP 2.3.8 einsetzen, wenn Sie zu Microsoft
kompatible MSCHAPv2/MPPE-Authentifikation und Verschlüsselung haben
wollen. Der Grund dafür ist, dass der aktuelle MSCHAPv2/MPPE-Patch
(19990813) gegen PPP 2.3.8 erstellt wurde. Wenn Sie keine zu
Microsoft kompatible Authentifikation und Verschlüsselung brauchen,
können Sie jede PPP-Quelle mit der Version 2.3.x verwenden.
```

Allerdings müssen Sie auf den Kernel einen Patch anwenden, der im Paket `kernel-patch-mppe` enthalten ist. Er stellt das Module pp_mppe für den pppd zur Verfügung.

Beachten Sie, dass Verschlüsselung in ppptp erfordert, dass Sie die Nutzerpasswörter in Klartext speichern. Außerdem sind für das MS-CHAPv2-Protokoll Sicherheitslücken bekannt (`http://mopo.informatik.uni-freiburg.de/pptp_mschapv2/`).

8.6 Infrastruktur für öffentliche Schlüssel (Public Key Infrastructure, PKI)

Mit der Infrastruktur für öffentliche Schlüssel (PKI) wurde eine Sicherheitsarchitektur eingeführt, um den Grad der Vertrauenswürdigkeit von Informationen, die über unsichere Netzwerke ausgetauscht werden, zu erhöhen. Sie beruht auf dem Konzept von öffentlichen und privaten kryptographischen Schlüsseln, um die Identität des Absenders (Signierung) zu überprüfen und die Geheimhaltung zu sichern (Verschlüsselung).

Wenn Sie über die Einrichtung einer PKI nachdenken, sehen Sie sich mit einer breiten Palette von Problemen konfrontiert:

- eine Zertifizierungsstelle (Certification Authority, CA), die Zertifikate ausgeben und bestätigen und unter einer bestimmten Hierarchie arbeiten kann
- ein Verzeichnis, das die öffentlichen Zertifikate der Benutzer enthält
- eine Datenbank (?), um eine List von Widerrufen von Zertifikaten (Certificate Revocation Lists, CRL) zu verwalten
- Geräte, die mit der CA zusammenarbeiten, um Smartcards/USB-Token oder ähnliches zu erzeugen und die Zertifikate sicher zu speichern
- Anwendungen, die die von einer CA ausgestellten Zertifikate benutzen können, um verschlüsselte Kommunikation zu aufzubauen und bestimmte Zertifikate gegen die CRL zu prüfen (zur Authentifizierung und so genannte “full Single Sign On solutions”)

- eine Zeitstempel-Autorität, um Dokumente digital zu signieren
- eine Verwaltungskonsole, von der aus dies alles vernünftig benutzt werden kann (Erstellung von Zertifikaten, Kontrolle der Widerruflisten, usw., . . .)

Debian GNU/Linux beinhaltet Softwarepaket, die Ihnen bei einigen dieser PKI-Probleme helfen können. Dazu gehört `OpenSSL` (zur Erstellung von Zertifikaten), `OpenLDAP` (für ein Verzeichnis, um die Zertifikate zu speichern) `gnupg` und `openswan` (mit X.509 Unterstützung). Jedoch stellt Debian zum Zeitpunkt der Veröffentlichung von Woody (Debian 3.0) keine der frei verfügbaren Certificate Authorities wie zum Beispiel pyCA, OpenCA (`http://www.openca.org`) oder die CA-Muster von OpenSSL zur Verfügung. Für weitere Informationen lesen Sie bitte das Open PKI Buch (`http://ospkibook.sourceforge.net/`).

8.7 SSL Infrastruktur

Debian stellt einige SSL-Zertifikate innerhalb der Distribution zur Verfügung, so dass Sie sie lokal installieren können. Sie befinden sich im Paket `ca-certificates`. Dieses Paket stellt eine zentrale Sammelstelle für Zertifikate dar, die an Debian übermittelt und vom Paketverwalter gebilligt (das heißt, verifiziert) wurden. Sie können für alle OpenSSL-Anwendungen, die SSL-Verbindungen verifizieren, nützlich sein.

FIXME: read debian-devel to see if there was something added to this.

8.8 Antiviren-Programme

Es gibt nicht viele Antiviren-Programme in Debian, wahrscheinlich weil die Benutzer von GNU/Linux nicht von Viren betroffen sind. Das Sicherheitsmodell von Unix trifft eine Unterscheidung zwischen privilegierten Prozessen (Root) und den Prozessen der Benutzer. Daher kann ein “feindliches” Programm, das ein Benutzer empfängt oder erstellt und dann ausführt, nicht das System “infizieren” oder daran Veränderungen vornehmen. Es existieren dennoch Würmer und Viren für GNU/Linux, auch wenn es (bisher) keinen Virus gab, der sich im Freien weit über eine Debian-Distribution verbreitet hat. Wie dem auch sei, Administratoren sollten vielleicht Antiviren-Gateways aufbauen, um verwundbarere Systeme in ihrem Netzwerk vor Viren zu schützen.

Debian GNU/Linux bietet derzeit die folgenden Werkzeuge zum Erstellen von Antiviren-Umgebungen an:

- Clam Antivirus (`http://www.clamav.net`), das in Debian *Sarge* (der aktuellen 3.1 Veröffentlichung) enthalten ist. Es sind Pakete sowohl für den Virusscanner (`clamav`) des Scanner-Daemons (`clamav-daemon`) als auch für die Daten, die der Scanner benötigt, verfügbar. Da es für die richtig Arbeit eines Antivirus-Programms entscheidend ist, dass seine Daten auf dem neusten Stand sind, gibt es zwei verschiedene Weg, um diese Daten

aktuell zu halten: `clamav-freshclam` eröffnet die Möglichkeit, die Datenbank automatisch über das Internet zu aktualisieren, und `clamav-data` stellt die Daten unmittelbar zur Verfügung.[2]

- `mailscanner` ist ein Gateway-Scanner, der in E-Mails Viren und Spam entdeckt. Er arbeitet auf der Grundlage von `sendmail` oder `exim` und kann mehr als 17 verschiedene Virensuch-Engines (einschließlich `clamav`) verwenden.
- `libfile-scan-perl`, welches File::Scan liefert. Das ist eine Erweiterung von Perl, mit der Dateien nach Viren durchsucht werden können. Mit diesem Modul können plattformunabhängige Virenscanner realisiert werden.
- Amavis Next Generation (`http://www.sourceforge.net/projects/amavis`) ist im Paket `amavis-ng` enthalten und in *Sarge* verfügbar. Es ist ein Virusscanner, der in verschiedene MTAs (Exim, Sendmail, Postfix oder Qmail) integriert werden kann. Er unterstützt mehr als 15 Virensuch-Engines (einschließlich clamav, File::Scan und openantivirus).
- sanitizer (`http://packages.debian.org/sanitizer`), ein Werkzeug, das das Paket `procmail` verwendet. Es kann den Anhang von E-Mails nach Viren durchsuchen, Anhänge aufgrund ihres Dateinamens abweisen und vieles mehr.
- amavis-postfix (`http://packages.debian.org/amavis-postfix`), ein Skript, das eine Schnittstelle vom Mail-Transport-Agent zu einem oder mehreren kommerziellen Viren-Scannern anbietet (dieses Paket ist lediglich für den MTA `postfix` bestimmt).
- `exiscan`, ein Virusscanner für E-Mails, der in Perl geschrieben wurde. Er arbeitet mit Exim zusammen.
- `sanitizer`, ein E-Mail-Scanner, der potenziell gefährliche Anhänge entfernen kann.
- `blackhole-qmail` ist ein Spamfilter für Qmail mit eingebauter Unterstützung von Clamav.

Einige Gateway-Daemons bieten schon Programmerweiterungen an, um Antiviren-Umgebungen zu erstellen. Dazu gehören `exim4-daemon-heavy` (die *heavy* Version des Exim MTAs), `frox`, ein transparenter caching FTP-Proxyserver), `messagewall` (ein SMTP-Proxyserver) und `pop3vscan` (ein transparenter POP3-Proxy).

Wie Sie sehen, enthält Debian selbst derzeit keine Antiviren-Software in der offiziellen Hauptdistribution (3.0 als dies geschrieben wurde). Es enthält aber verschiedene Schnittstellen, um

[2] Wenn Sie das letztere Paket verwenden und ein offizielles Debian betreiben, wird die Datenbank nicht im Zuge von Sicherheitsaktualisierung auf den neusten Stand gebracht. Sie sollten entweder `clamav-freshclam`, `clamav-getfiles` verwenden, um neue `clamav-data`-Pakete zu erstellen, oder die Datenbank über die Seite der Betreuer aktuell halten:

```
deb http://people.debian.org/~zugschlus/clamav-data/ / deb-src http://people.debian.org/~zugschlus
/
```

Gateways mit Antivirenfunktionen zu erstellen. Der Scanner `clamav` wird in der nächsten offiziellen Veröffentlichung enthalten sein.

Im Folgenden einige andere freie Antiviren-Projekte, die in der Zukunft in Debian enthalten sein könnten:

- Open Antivirus (http://sourceforge.net/projects/openantivirus/) (siehe Fehler #150698 (ITP oav-scannerdaemon) (http://bugs.debian.org/150698) und Fehler #150695 (ITP oav-update) (http://bugs.debian.org/150695)).

FIXME: Is there a package that provides a script to download the latest virus signatures from http://www.openantivirus.org/latest.php?

FIXME: Check if scannerdaemon is the same as the open antivirus scanner daemon (read ITPs).

Allerdings wird Debian *niemals* kommerzielle Antiviren-Software anbieten. Dazu zählen Panda Antivirus NAI Netshield, Sophos Sweep (http://www.sophos.com/), TrendMicro Interscan (http://www.antivirus.com) oder RAV (http://www.ravantivirus.com). Weitere Hinweise finden Sie im Mini-FAQ 'Antiviren-Software für Linux/Unix' (http://www.computer-networking.de/~link/security/av-linux.txt). Das bedeutet nicht, dass diese Software auf einem Debian-System richtig installiert werden kann.

Zusätzliche Informationen über das Aufsetzen eines Systems zur Virenerkennung erhalten Sie im Artikel Building an E-mail Virus Detection System for Your Network (http://www.linuxjournal.com/article.php?sid=4882) von Dave Jones.

8.9 GPG-Agent

Es ist heutzutage weit verbreitet, E-Mails digital zu unterschreiben (manchmal auch zu verschlüsseln). Sie können z.B. feststellen, dass viele Menschen auf Mailinglisten ihre E-Mails signieren. Signaturen von öffentlichen Schlüsseln ist im Moment die einzige Möglichkeit um festzustellen, ob eine E-Mail vom Absender geschickt wurden und nicht von jemand anderem.

Debian GNU/Linux enthält eine Anzahl von E-Mail-Clients mit der eingebauten Fähigkeit, E-Mails zu signieren. Sie arbeiten entweder mit `gnupg` oder `pgp` zusammen:

- `evolution`.
- `mutt`.
- `kmail`.
- `mozilla` oder Thunderbird (im Paket `mozilla-thunderbird` enthalten), falls das Enigmail (http://enigmail.mozdev.org/)-Plugin installiert ist, was durch das Paket `mozilla-enigmail` und `mozilla-thunderbird-enigmail` bereitgestellt wird.
- `sylpheed`. Abhängig davon wie sich die stabile Version dieses Pakets entwickelt, müssen Sie die *bleeding edge* Version, `sylpheed-claws`, verwenden.

- `gnus` wird mit dem Paket `mailcrypt` installiert und ist eine Schnittstelle für `emacs` zu `gnupg`.
- `kuvert` stellt diese Funktion unabhängig von Ihrem Mail-User-Agent (MUA) zur Verfügung, indem es mit dem Mail-Transport-Agent (MTA) arbeitet.

Key-Server ermöglichen es Ihnen, veröffentlichte öffentliche Schlüssel herunterzuladen, damit Sie Signaturen überprüfen können. Einer diese Key-Server ist `http://wwwkeys.pgp.net`. `gnupg` kann automatisch öffentliche Schlüssel holen, die sich nicht schon in Ihrem öffentlichen Schlüsselbund befinden. Um beispielsweise `gnupg` so einzurichten, dass es den oben genannten Key-Server verwendet, müssen Sie die Datei `~/.gnupg/options` bearbeiten und folgende Zeile hinzufügen: [3]

```
keyserver wwwkeys.pgp.net
```

Die meisten Key-Server sind miteinander verbunden. Wenn Sie also Ihren öffentlichen Schlüssel einem hinzufügen, wird er an alle anderen Key-Server weitergereicht. Da wäre auch noch das Debian GNU/Linux Paket `debian-keyring`, das die öffentlichen Schlüssel aller Debian-Entwickler enthält. Der Schlüsselbund von `gnupg` wird in `/usr/share/keyrings/` installiert.

Weitere Informationen:

- GnuPG FAQ (`http://www.gnupg.org/faq.html`).
- Das GNU-Handbuch zum Schutze der Privatsphäre (`http://www.gnupg.org/gph/de/manual/index.html`).
- Gnu Privacy Guard (GnuPG) Mini Howto (`http://www.dewinter.com/gnupg_howto/german/GPGMiniHowto.html`).
- comp.security.pgp FAQ (`http://www.uk.pgp.net/pgpnet/pgp-faq/`).
- Keysigning-Party HOWTO (`http://www.cryptnet.net/fdp/crypto/gpg-party/gpg-party.de.html`).

[3] Weitere Beispiele wie Sie `gnupg` konfigurieren können, finden Sie in `/usr/share/doc/mutt/examples/gpg.rc`.

Kapitel 9

Der gute Umgang von Entwicklern mit der Sicherheit des OS

Dieses Kapitel handelt von einigen der anerkannten Vorgehensweisen, wenn Entwickler Pakete für Debian erstellen. Wenn Sie sehr an sicherheitsbewusster Programmierung interessiert sind, sollten Sie David Wheelers Secure Programming for Linux and Unix HOWTO (http://www.dwheeler.com/secure-programs/) und Secure Coding: Principles and Practices (http://www.securecoding.org) von Mark G. Graff und Kenneth R. van Wyk (O'Reilly, 2003) lesen.

9.1 Das richtige Vorgehen für die Nachprüfung der Sicherheit und Gestaltung

Entwickler, die Software in Pakete packen, sollten größte Anstrengung darauf verwenden sicherzustellen, dass die Installation der Software und ihre Verwendung keine Sicherheitsrisiken für das System oder seine Benutzer eröffnet.

Dazu sollten sie vor der Veröffentlichung der Software oder einer neuen Version den Quellcode des Pakets nachprüfen, um Fehler zu finden, die zu Sicherheitslücken führen können. Bekanntermaßen ist der Aufwand für die Fehlerbehebung in verschiedenen Stadien der Entwicklung unterschiedlich. So ist es leichter (und billiger), Fehler während der Entwicklung auszubessern als später, wenn die Software schon herausgegeben wurde und nur noch gewartet wird (einige Studien behaupten, dass die Kosten in dieser Phase **60** Mal höher sind). Es gibt Hilfsmittel, die versuchen, Fehler automatisch zu entdecken. Entwickler sollten dennoch die verschiedenen Sicherheitsfehler kennen, damit sie sie verstehen und sie so in eigenen (oder fremden) Programmcode entdecken können.

Programmierfehler, die typischerweise zu Sicherheitsproblemen führen, sind insbesondere: Pufferüberläufe, (http://de.wikipedia.org/wiki/Puffer%C3%BCberlauf), Format-String-Überläufe, Heap-Überläufe und Integer-Überläufe (in C/C++-Programmen), vorübergehende Symlink-Schwachstellen (http://de.wikipedia.org/wiki/

`Symlink-Schwachstelle`) (in Skripten), Directory Traversal (http://de.wikipedia.org/wiki/Directory_Traversal), die Einschleusung von Befehlen (auf Servern) und Cross-Site Scripting (http://de.wikipedia.org/wiki/Cross-Site_Scripting) sowie SQL-Injektionen (http://de.wikipedia.org/wiki/SQL-Injektion) (bei web-orientierten Anwendungen). Eine ausführliche Liste von Sicherheitsfehlern finden Sie in Fortifys Taxonomy of Software Security Errors (http://vulncat.fortifysoftware.com/).

Einige dieser Probleme können Sie nur erkennen, wenn Sie ein Experte in der verwendeten Programmiersprache sind. Aber andere können leicht entdeckt und behoben werden. Zum Beispiel kann eine Symlink-Schwachstelle auf Grund einer falschen Verwendung von temporären Verzeichnissen ohne Weiteres entdeckt werden, indem Sie `grep -r "/tmp/" .` ausführen. Diese Verweise sollten überprüft werden und fest einprogrammierte Dateinamen in temporären Verzeichnissen in Shell-Skripten mit `mktemp` oder `tempfile`, in Perl-Skripten mit `File::Temp(3perl)` und in C/C++ mit `tmpfile(3)` ersetzt werden.

Es stehen Ihnen einige Werkzeuge zur Verfügung, die Sie dabei unterstützen, den Quellcode auf Sicherheitsprobleme hin zu überprüfen. Dazu zählen `rats`, `flawfinder` und `pscan`. Weitere Informationen finden Sie in der Liste der Werkzeuge, die vom Debian-Security-Audit-Team verwendet werden (http://www.de.debian.org/security/audit/tools).

Beim Paketieren von Software sollten Entwickler darauf achten, dass sie allgemein anerkannte Sicherheitsprinzipien einhalten. Dazu gehören:

- Die Software sollte mit so geringen Rechten wie möglich laufen
  - Falls das Paket Binaries mit setuid oder setgid enthält, wird `Lintian` vor setuid (http://lintian.debian.org/reports/Tsetuid-binary.html)-, setgid (http://lintian.debian.org/reports/Tsetgid-binary.html)- und setuid und setgid (http://lintian.debian.org/reports/Tsetuid-gid-binary.html)-Binaries warnen.
  - Die Daemons, die in einem Paket enthalten sind, sollten mit den Rechten eines Benutzers laufen, der nur geringe Privilegien besitzt (vergleichen Sie dazu 'Benutzer und Gruppen für Daemons erstellen' auf der nächsten Seite).
- Automatisierte Aufgaben (also mit `cron`) sollten NICHT mit Root-Rechten laufen. Zumindest sollten mit Root-Rechten keine komplizierten Aufgaben erledigt werden.

Falls Sie diese Prinzipien nicht einhalten können, sollten Sie sichergehen, dass das Programm, das mit umfangreicheren Rechten läuft, auf Sicherheitsprobleme überprüft wurde. Wenn Sie sich nicht sicher sind oder Hilfe benötigen, sollten Sie sich mit dem Sicherheits-Audit-Team von Debian (http://www.de.debian.org/security/audit/) in Verbindung setzen. Wenn Binaries setuid/setgid verwenden, sollten Sie die Richtlinie von Debian zu Rechten und Besitzern (http://www.debian.org/doc/debian-policy/ch-files#s10.9) beachten.

Für weitere Informationen, insbesondere hinsichtlich Sicherheitsfragen, sollten Sie das Secure Programming for Linux and Unix HOWTO (http://www.dwheeler.com/secure-programs/) und das Build Security In (https://buildsecurityin.us-cert.gov/portal/) Portal lesen (oder den Programmautor darauf hinweisen).

9.2 Benutzer und Gruppen für Daemons erstellen

Wenn Ihre Software als Daemon läuft, der keine Root-Rechte benötigt, müssen Sie für ihn einen Benutzer erstellen. Es gibt zwei Arten von Benutzern in Debian, die für Pakete verwendet werden können: statische UIDs (werden von `base-passwd` vergeben, eine Liste der statischen Benutzern in Debian finden Sie bei 'Benutzer und Gruppen des Betriebssystems' auf Seite 204) und dynamisches UIDs, die in einem zugewiesenen Bereich liegen.

Im ersten Fall müssen Sie mit `base-passwd` eine Benutzer- oder Gruppen-ID erstellen. Wenn der Benutzer verfügbar ist, muss das Paket, das Sie anbieten möchten, eine Abhängigkeit vom Paket `base-passwd` enthalten.

Im zweiten Fall müssen Sie den Systembenutzer entweder entweder *preinst* oder *postinst* erstellen und dafür sorgen, dass das Paket von `adduser (>= 3.11)` abhängt.

Im folgenden Programmbeispiel soll gezeigt werden, wie der Benutzer oder Gruppe, mit deren Rechten der Daemon laufen wird, bei der Installation oder Aktualisierung des Pakets erstellt wird.

```
[...]
case "$1" in
  install|upgrade)

  # If the package has default file it could be sourced, so that
  # the local admin can overwrite the defaults

  [ -f "/etc/default/packagename" ] && . /etc/default/packagename

  # Sane defaults:

  [ -z "$SERVER_HOME" ] && SERVER_HOME=server_dir
  [ -z "$SERVER_USER" ] && SERVER_USER=server_user
  [ -z "$SERVER_NAME" ] && SERVER_NAME="Server description"
  [ -z "$SERVER_GROUP" ] && SERVER_GROUP=server_group

  # Groups that the user will be added to, if undefined, then none.
  ADDGROUP=""

  # create user to avoid running server as root
  # 1. create group if not existing
  if ! getent group | grep -q "^$SERVER_GROUP:" ; then
     echo -n "Adding group $SERVER_GROUP.."
     addgroup --quiet --system $SERVER_GROUP 2>/dev/null ||true
     echo "..done"
  fi
  # 2. create homedir if not existing
  test -d $SERVER_HOME || mkdir $SERVER_HOME
```

```
    # 3. create user if not existing
    if ! getent passwd | grep -q "^$SERVER_USER:"; then
      echo -n "Adding system user $SERVER_USER.."
      adduser --quiet \
              --system \
              --ingroup $SERVER_GROUP \
              --no-create-home \
              --disabled-password \
              $SERVER_USER 2>/dev/null || true
      echo "..done"
    fi
    # 4. adjust passwd entry
    usermod -c "$SERVER_NAME" \
            -d $SERVER_HOME   \
            -g $SERVER_GROUP  \
               $SERVER_USER
    # 5. adjust file and directory permissions
    if ! dpkg-statoverride --list $SERVER_HOME >/dev/null
    then
        chown -R $SERVER_USER:adm $SERVER_HOME
        chmod u=rwx,g=rxs,o= $SERVER_HOME
    fi
    # 6. Add the user to the ADDGROUP group
    if test -n $ADDGROUP
    then
        if ! groups $SERVER_USER | cut -d: -f2 | \
           grep -qw $ADDGROUP; then
             adduser $SERVER_USER $ADDGROUP
        fi
    fi
    ;;
    configure)

  [...]
```

Außerdem müssen Sie für das Init.d-Skript sicherstellen,

- dass der Daemon beim Starten seine Rechte ablegt: Wenn die Software nicht selbst den `setuid(2)` oder `seteuid(2)` Aufruf absetzt, sollten Sie die Option `--chuid` für `start-stop-daemon` verwenden.
- dass der Daemon nur angehalten wird, wenn die Benutzer-IDs übereinstimmen. Dafür ist die Option `--user` von `start-stop-daemon` hilfreich.
- dass der Daemon nicht gestartet wird, wenn sein Benutzer oder Gruppe nicht existiert:

```
if ! getent passwd | grep -q "^server_user:"; then
   echo "Server user does not exist. Aborting" >&2
   exit 1
fi
if ! getent group | grep -q "^server_group:" ; then
   echo "Server group does not exist. Aborting" >&2
   exit 1
fi
```

Wenn das Paket einen Systembenutzer erstellt, kann er wieder in *postrm* entfernt werden, wenn das Paket vollständig gelöscht wird (purge). Dabei gibt es allerdings einen Nachteil. Zum Beispiel werden Dateien, die von dem Benutzer des Daemons erstellt wurden, benutzerlos und können später einem neuen Benutzer gehören, dem die gleiche UID zugewiesen wurde [1]. Daher ist nicht zwingend notwendig, dass Benutzer beim vollständigen Löschen eines Pakets entfernt werden. Dies hängt vielmehr vom jeweiligen Paket ab. Im Zweifelsfall sollte der Administrator gefragt werden (mit `debconf`), was passieren soll, wenn ein Paket gelöscht wird.

Sehen Sie sich folgenden Code an [2], der zuvor erstellte Benutzer und Gruppen entfernt. Dies geschieht aber nur dann, wenn die UID im Bereich der dynamisch zugewiesenen System-UIDs liegt und die GID einer Systemgruppe angehört:

```
case "$1" in
 purge)
[...]
   # find first and last SYSTEM_UID numbers
   for LINE in ‘grep SYSTEM_UID /etc/adduser.conf | grep -v "^#"‘; do
      case $LINE in
         FIRST_SYSTEM_UID*)
           FIST_SYSTEM_UID=‘echo $LINE | cut -f2 -d ’=‘‘
           ;;
         LAST_SYSTEM_UID*)
           LAST_SYSTEM_UID=‘echo $LINE | cut -f2 -d ’=‘‘
           ;;
         *)
           ;;
         esac
   done
   # Remove system account if necessary
   CREATEDUSER="server_user"
   if [ -n "$FIST_SYSTEM_UID" ] && [ -n "$LAST_SYSTEM_UID" ]; then
```

[1] Interessante Diskussionen zu diesem Thema finden sich in http://lists.debian.org/debian-mentors/2004/10/msg00338.html und http://lists.debian.org/debian-devel/2004/05/msg01156.html.

[2] Unter Umständen wird er als `dh_adduser` in debhelper enthalten sein. Sehen Sie sich dazu #81967 (http://bugs.debian.org/81697), #291177 (http://bugs.debian.org/291177) und #118787 (http://bugs.debian.org/118787) an.

```
    if USERID=‘getent passwd $CREATEDUSER | cut -f 3 -d ’:’‘; then
      if [ -n "$USERID" ]; then
        if [ "$FIST_SYSTEM_UID" -le "$USERID" ] && \
           [ "$USERID" -le "$LAST_SYSTEM_UID" ]; then
             echo -n "Removing $CREATEDUSER system user.."
             deluser --quiet $CREATEDUSER || true
             echo "..done"
        fi
      fi
    fi
  fi
  # Remove system group if necessary
  CREATEDGROUP=server_group
  FIRST_USER_GID=‘grep ^USERS_GID /etc/adduser.conf | cut -f2 -d ’=’‘
  if [ -n "$FIST_USER_GID" ] then
    if GROUPGID=‘getent group $CREATEDGROUP | cut -f 3 -d ’:’‘; then
      if [ -n "$GROUPGID" ]; then
        if [ "$FIST_USER_GID" -gt "$GROUPGID" ]; then
          echo -n "Removing $CREATEDGROUP group.."
          delgroup --only-if-empty $CREATEDGROUP || true
          echo "..done"
        fi
      fi
    fi
  fi
[...]
```

Wenn ein Programm unter einem Benutzer mit beschränkten Rechten läuft, wird sichergestellt, dass Sicherheitsprobleme nicht das gesamte System beschädigen können. Dieses Vorgehen beachtet auch das Prinzip der *geringst möglichen Privilegien*. Denken Sie daran, dass Sie die Rechte eines Programms auch noch durch andere Methoden als beschränkte Benutzerrechte weiter einschränken können [3]. Weitere Informationen finden Sie im Abschnitt Minimize Privileges (http://www.dwheeler.com/secure-programs/Secure-Programs-HOWTO/minimize-privileges.html) des Buchs *Secure Programming for Linux and Unix HOWTO*.

[3] Sie können sogar eine SELinux-Richtlinie erstellen.

Kapitel 10

Vor der Kompromittierung

10.1 Halten Sie Ihr System sicher

Sie sollten bestrebt sein, Ihr System sicher zu halten, indem Sie seine Verwendung und die es betreffenden Verwundbarkeiten im Auge behalten. Sobald Patches verfügbar sind, sollte Sie diese auch einspielen. Denn auch wenn Sie zu Beginn ein sehr sicheres System eingerichtet haben, sollten Sie sich erinnern, dass die Sicherheit eines Systems mit der Zeit nachlässt. Das liegt daran, dass Sicherheitslücken in Systemdiensten entdeckt werden können. Außerdem können Benutzer die Sicherheit untergraben, wenn ihnen das notwendige Verständnis fehlt (z.B. wenn sie entfernt auf ein System mit einem Klartextpasswort oder einem einfach zu erratenden Passwort zugreifen) oder gar weil sie aktiv versuchen, die Sicherheit des Systems auszuschalten (indem Sie z.B. zusätzliche Dienste lokal in ihren Konten installieren).

10.1.1 Beobachtung von Sicherheitslücken

Die meisten Administratoren werden sich Sicherheitslücken, die ihr System betreffen, bewusst, wenn sie den dazugehörigen Patch sehen. Sie können aber Angriffen schon im Vorfeld begegnen und vorübergehende Abwehrmaßnahmen einleiten, sobald Sie festgestellt haben, dass Ihr System verwundbar ist. Dies gilt besonders für exponierte Systeme (sind z.B. mit dem Internet verbunden), die Dienste anbieten. In diesem Fall sollte der Systemadministrator einen Blick auf die bekannten Informationsquellen werfen, um als erster zu wissen, wenn eine Sicherheitslücke für einen kritischen Dienst entdeckt wird.

Typischerweise abonniert man eine Mailingliste für Ankündigungen und beobachtet Webseiten oder Fehlerverfolgungssysteme der Software-Entwickler eines bestimmten Programms. So sollten beispielsweise Apache-Benutzer regelmäßig Apaches Auflistung von Sicherheitslücken (`http://httpd.apache.org/security_report.html`) durchsehen und die Mailingliste Ankündigungen für den Apache-Server (`http://httpd.apache.org/lists.html#http-announce`) abonnieren.

Um bekannte Sicherheitslücken, die die Debian-Distribution betreffen, zu verfolgen, bietet das Security-Testing-Team von Debian einen Sicherheitstracker (`http:`

//security-tracker.debian.net/) an. Dieser führt alle bekannten Sicherheitslücken auf, die in Paketen von Debian noch nicht ausgebessert wurden. Die Informationen im Tracker stammen aus öffentlich zugänglichen Quellen. Dazu zählen Datenbanken über Sicherheitslücken und die Fehlerdatenbank von Debian (http://www.debian.org/Bugs/). Administratoren können nach bekannten Sicherheitsmängeln für Stable (http://security-tracker.debian.net/tracker/status/release/stable), Oldstable (http://security-tracker.debian.net/tracker/status/release/oldstable), Testing (http://security-tracker.debian.net/tracker/status/release/testing) oder Unstable (http://security-tracker.debian.net/tracker/status/release/unstable) suchen.

Der Tracker kann mittels einer Benutzerschnittstelle durchsucht werden (nach CVE (http://cve.mitre.org/)-Namen und dem Paketnamen). Einige Werkzeuge (wie zum Beispiel debsecan (http://packages.debian.org/debsecan)) setzen diese Datenbank ein, um auf Verwundbarkeiten des betreffenden Systems hinzuweisen, die noch nicht ausgebessert wurden (d.h. für die eine Ausbesserung bevorsteht).

Sicherheitsbewusste Administratoren können mit diesen Informationen feststellen, welche Sicherheitslücken das System, das sie verwalten, betreffen könnten, wie schwer das Risiko der Lücke wiegt und ob vorübergehend Gegenmaßnahmen zu treffen sind (falls möglich), bis ein Patch verfügbar ist, der das Problem löst.

Sicherheitsprobleme in Veröffentlichungen, die vom Sicherheitsteam von Debian unterstützt werden, sollten irgendwann in Debian-Sicherheits-Ankündigungen (DSA) behandelt werden, die allen Benutzern zur Verfügung gestellt werden (vergleiche 'Fortlaufende Aktualisierung des Systems' auf dieser Seite). Sobald ein Sicherheitsproblem ausgebessert wurde und die Lösung in einer Ankündigung enthalten ist, wird es nicht mehr im Tracker aufgeführt. Sie können es aber immer noch mit einer Suchanfrage (nach dem CVE-Namen) finden, indem Sie Querverweise für Debian-Sicherheitsankündigungen (http://www.de.debian.org/security/crossreferences) verwenden.

Beachten Sie aber, dass die Informationen im Tracker des Debian-Sicherheitsteams nur bekannte Sicherheitslücken (d.h. solche, die öffentlich sind) beinhaltet. In einigen Fällen gibt das Debian-Sicherheitsteam DSA für Pakete heraus, die auf vertraulichen Informationen beruhen, die das Team erhalten hat (z.B. über nicht-öffentliche Mailinglisten der Distributionen oder von Programmautoren). Seien Sie also nicht überrascht, in Sicherheitsankündigungen Sicherheitsprobleme zu entdecken, die nicht im Tracker enthalten sind.

10.1.2 Fortlaufende Aktualisierung des Systems

Sie sollten regelmäßig Sicherheitsaktualisierungen durchführen. Der ganz überwiegende Anteil der Exploits nutzt bekannte Sicherheitslücken aus, die nicht rechtzeitig ausgebessert wurden. Dies wird in der Veröffentlichung von Bill Arbaugh (http://www.cs.umd.edu/~waa/vulnerability.html) dargestellt, die 2001 auf dem IEEE Symposium on Security and Privacy vorgestellt wurde. Das Durchführen einer Aktualisierung wird unter 'Ausführen von Sicherheitsupdates' auf Seite 48 beschrieben.

Überprüfung von Hand, welche Sicherheitsaktualisierungen verfügbar sind

Debian besitzt ein Werkzeug, um zu überprüfen, ob ein System aktualisiert werden muss (siehe Tiger unten). Viele Nutzer wollen aber einfach von Hand überprüfen, ob Sicherheitsaktualisierungen für ihr System zur Verfügung stehen.

Wenn Sie Ihr System nach der Beschreibung unter 'Ausführen von Sicherheitsupdates' auf Seite 48 eingerichtet haben, müssen Sie nur Folgendes tun:

```
# apt-get update
# apt-get upgrade -s
[ ... überprüfen der zu aktualisierenden Pakete ... ]
# apt-get upgrade
# checkrestart
[ ... Neustart der Dienste, die neu gestartet werden müssen ... ]
```

Weiter müssen alle Dienste, deren Bibliotheken aktualisiert wurden, neu gestartet werden. Bemerkung: Lesen Sie 'Ausführen von Sicherheitsupdates' auf Seite 48 für weitere Informationen zu Bibliotheks- (und Kernel-)Aktualisierungen.

Die erste Zeile wird die Liste der verfügbaren Pakete von den festgelegten Paketquellen herunterladen. Die Option `-s` wird eine Simulation durchführen, d.h. es werden *keine* Pakete heruntergeladen oder installiert. Vielmehr teilt es Ihnen mit, welche heruntergeladen und installiert werden sollen. Durch dieses Ergebnis könnten Sie erfahren, welche Pakete von Debian ausgebessert wurden und als Sicherheitsaktualisierung verfügbar sind. Zum Beispiel:

```
# apt-get upgrade -s
Reading Package Lists... Done
Building Dependency Tree... Done
2 packages upgraded, 0 newly installed, 0 to remove and 0  not upgraded.
Inst cvs (1.11.1p1debian-8.1 Debian-Security:3.0/stable)
Inst libcupsys2 (1.1.14-4.4 Debian-Security:3.0/stable)
Conf cvs (1.11.1p1debian-8.1 Debian-Security:3.0/stable)
Conf libcupsys2 (1.1.14-4.4 Debian-Security:3.0/stable)
```

In diesem Beispiel können Sie erkennen, dass auf dem System `cvs` und `cupsys` mit neuen Versionen aus *Woodys* oder *Sarges* Sicherheitsarchiv aktualisiert werden müssen. Um herauszufinden, warum eine Aktualisierung notwendig ist, sollten Sie `http://security.debian.org` besuchen und sich ansehen, welche aktuellen Debian-Sicherheits-Ankündigungen zu diesen Paketen veröffentlicht wurden. In unserem Fall sind die zugehörigen DSA DSA-233 (`http://www.debian.org/security/2003/dsa-233`) (für `cvs`) und DSA-232 (`http://www.debian.org/security/2003/dsa-232`) (für `cupsys`).

Es ist zu beachten, dass der Rechner neugestartet werden muss, wenn der Kernel aktualisiert wurde.

Automatisches Überprüfung von Aktualisierungen mit cron-apt

Eine andere Methode für automatische Sicherheitsaktualisierungen ist die Verwendung von `cron-apt`. Dieses Paket stellt ein Werkzeug zur Verfügung, mit dem das System in regelmäßigen Abständen (mit einem Cronjob) aktualisiert wird. Standardmäßig wird es die Paketliste aktualisieren und neue Pakete herunterladen. Es kann auch so konfiguriert werden, dass es Mails an den Systemadministrator schickt.

Hinweis: Wenn Sie vorhaben, Ihr System automatisch zu aktualisieren (auch wenn Sie sich nur die Pakete herunterladen), sollten Sie sich vielleicht die Distributionsversion ansehen, wie in 'Überprüfung der Distribution mit der `Release`-Datei' auf Seite 147 beschrieben wird. Anderenfalls können Sie sich nicht sicher sein, dass die heruntergeladenen Pakete wirklich aus einer vertrauenswürdigen Quelle stammen.

Verwendung von Tiger, um automatisch Sicherheitsaktualisierungen zu überprüfen

Wenn Sie nach einem Programm suchen, das schnell die Verwundbarkeit des Systems überprüft und gefundene Sicherheitslücken meldet, sollten Sie das Paket `tiger` ausprobieren. Das Paket besteht aus einer Anzahl von Skripten für die Bourne-Shell, C-Programmen und Datendateien, die dazu verwendet werden, um Sicherheitsaudits durchzuführen. Das Paket in Debian GNU/Linux beinhaltet zusätzliche Erweiterungen, die auf die Debian-Distribution abgestimmt sind. Damit stehen mehr Funktionen zur Verfügung als in den Tigerskripten von TAMU (oder sogar von TARA, eine Tigerversion, die von ARSC vertrieben wird). Lesen Sie für weitere Informationen die Datei `README.Debian` und die Handbuchseite `tiger(8)`.

Eine dieser Verbesserungen ist das Skript `deb_checkadvisories`. Diese Skript verwendet eine Liste von DSAs und gleicht sie mit den installierten Paketen ab. Es meldete dann alle Pakete, die laut dem Debian Security Team verwundbar sind. Dies ist eine etwas andere, allgemeinere Herangehensweise als im Tigerskript `check_signatures`, das die MD5-Summen von Programmen mit bekannten Lücken testet.

Da Debian im Moment keine Liste der MD5-Summen von Programmen mit bekannten Lücken liefert (wie sie von anderen Betriebssystemen wie Sun Solaris verwendet wird), wird die *Überprüfung-der-DSAs-Herangehensweise* verwendet. Das Problem sowohl der Herangehensweise mit DSAs als auch der mit MD5-Summen ist, dass die Signaturen regelmäßig aktualisiert werden müssen.

Im Moment wird das dadurch gelöst, indem einen neue Version des Tigerpakets erstellt wird. Aber es steht nicht fest, dass der Paketbetreuer jedes Mal eine neue Version erstellt, wenn ein DSA bekannt gegeben wird. Eine nette Erweiterung, die aber noch nicht implementiert ist, wäre es, wenn das eigenständig durchgeführt wird. Das umfasst, dass die DSAs aus dem Netz heruntergeladen werden, eine Liste erstellt wird und dann die Prüfung durchgeführt wird. Die DSAs werden im Moment aus der lokalen CVS-Aktualisierung der WML-Quellen des Betreuers aktualisiert, die dazu verwendet werden, `http://security.debian.org` (der Webserver) zu erstellen.

Ein Programm wäre wünschenswert, das die DSAs, die per E-Mail empfangen wurden oder auf security.debian.org verfügbar sind, analysiert und dann die Datei erstellt, die von

`deb_checkadvisories` verwendet wird, um Verwundbarkeiten zu bestätigen. Schicken Sie es als einen Fehlerbericht von `tiger`.

Die erwähnte Überprüfung wird über die Standardkonfiguration des Programms ausgeführt, wenn sie einmal eingerichtet wurde (siehe `/etc/tiger/cronrc`):

```
# Check for Debian security measures every day at 1 AM
#
1 * *   deb_checkmd5sums deb_nopackfiles deb_checkadvisories
#
```

Es gibt noch eine zusätzliche Überprüfung, die Sie vielleicht hinzufügen sollten, und welche noch kein Bestandteil des Standard-`Cron`-Skripts ist. Diese Überprüfung ist das Skript `check_patches`, das auf folgende Art und Weise funktioniert:

- führt `apt-get update` aus.
- überprüft, ob neue Pakete verfügbar sind.

Wenn Sie ein *Stable*-System betreiben und Sie die `Apt`-Quellen security.debian.org in Ihre `/etc/apt/sources.list` eingetragen haben (wie in 'Ausführen von Sicherheitsupdates' auf Seite 48 beschrieben), wird dieses Skript Ihnen mitteilen können, ob neue Pakete verfügbar sind, die Sie installieren sollten. Da die einzigen Pakete, die sich bei dieser Einstellung verändern, Sicherheitsaktualisierungen sind, bekommen Sie genau das, was Sie wollen.

Das funktioniert natürlich nicht, wenn Sie *Testing* oder *Sid/Unstable* am Laufen haben, da wahrscheinlich neue Pakete zahlreicher sind als Sicherheitsaktualisierungen.

Sie können dieses Skript den Überprüfungen hinzufügen, die vom `Cron`-Job durchgeführt werden (in der obigen Konfigurationsdatei). Dadurch würde `tigercron` Mails mit neuen Paketen verschicken (an denjenigen, der von `Tiger_Mail_RCPT` in `/etc/tiger/tigerrc` bezeichnet wurde).

```
# Check for Debian security measures every day at 1 am
#
1 * *   deb_checkmd5sums deb_nopackfiles check_patches
#
```

Andere Methoden für Sicherheitsaktualisierungen

Sie sollten auch einen Blick auf secpack (`http://clemens.endorphin.org/secpack/`) werfen. Es ist ein inoffizielles Programm, um Sicherheitsaktualisierungen von security.debian.org mit Prüfung der Signatur durchzuführen. Es wurde von Fruhwirth Clemens geschrieben.

10.1.3 Vermeiden Sie den Unstable-Zweig

Falls Sie nicht Zeit darauf verwenden wollen, selbst Pakete zu patchen, wenn Verwundbarkeiten entdeckt werden, sollten Sie auf produktiven Systemen *nicht* Debians Unstable-Zweig verwenden. Der Hauptgrund dafür ist, dass es für *Unstable* keine Sicherheitsaktualisierungen gibt (siehe 'Wie wird die Sicherheit für Testing und Unstable gehandhabt?' auf Seite 219).

Es ist eine Tatsache, dass manche Sicherheitsprobleme nur in Unstable auftreten und *nicht* in *Stable*. Das rührt daher, dass dort ständig neue Funktionen zu den Anwendungen hinzugefügt werden und auch neue Anwendungen aufgenommen werden, die unter Umständen noch nicht vollständig getestet wurden.

Um im *Unstable*-Zweig Sicherheitsaktualisierungen durchzuführen, müssen Sie vielleicht eine vollständige Aktualisierung mit einer neuen Version durchführen (was viel mehr als nur das betroffene Pakete aktualisieren könnte). Sicherheitsaktualisierungen wurden - mit Ausnahmen - nur in den *Stable*-Zweig zurückportiert. Die Grundidee ist, dass zwischen den Aktualisierungen *kein neuer Code* hinzugefügt werden sollte, sondern nur Beseitigungen von wichtigen Problemen.

Denken Sie daran, dass Sie allerdings den Sicherheitstracker verwenden können (wie unter 'Beobachtung von Sicherheitslücken' auf Seite 179 beschrieben), um bekannte Sicherheitsprobleme für diesen Zweig nachzuvollziehen.

10.1.4 Sicherheitsunterstützung für den Testing-Zweig

Wenn Sie den *Testing*-Zweig verwenden, müssen Sie einige Problemkreise hinsichtlich der Verfügbarkeit von Sicherheitsaktualisierungen in Betracht ziehen:

- Wenn eine Sicherheitslücke geschlossen wurde, portiert das Security Team den Patch nach *Stable* zurück (da Stable normalerweise einige Minor- oder Majorversionen zurückliegt). Die Paketbetreuer sind dafür verantwortlich, Pakete für den *Unstable*-Zweig vorzubereiten. Grundlage dafür ist normalerweise eine neue Veröffentlichung des Originalprogramms. Manchmal ereignen sich die Änderungen fast zur selben Zeit, und manchmal enthält eine der Veröffentlichungen eine Ausbesserung einer Sicherheitslücke vor einer anderen. Pakete in *Stable* werden gründlicher getestet als die in *Unstable*, da letztere in den meisten Fällen die neueste Veröffentlichung des Originalprogramms enthält (welches neue, unbekannte Fehler enthalten könnte).

- Gewöhnlich sind Sicherheitsaktualisierungen für den *Unstable*-Zweig verfügbar, wenn der Paketbetreuer ein neues Paket baut, und für den *Stable*-Zweig, wenn das Security Team eine neue Version hochlädt und ein DSA veröffentlicht. Beachten Sie, dass beides nicht des *Testing*-Zweig verändert.

- Wenn keine (neuen) Fehler in der *Unstable*-Version des Pakets entdeckt werden, wandert es nach ein paar Tagen nach *Testing*. Das dauert normalerweise zehn Tage. Es hängt allerdings von der Priorität des Hochladens der Veränderung ab und davon, ob das Paket

von *Testing* zurückgehalten wird, da Abhängigkeiten nicht aufgelöst werden können. Beachten Sie, dass wenn das Paket daran gehindert ist, nach Testing zu wandern, auch die Priorität des Hochladens daran nichts ändern kann.

Dieses Verhalten könnte sich je nach dem Status der Veröffentlichung der Distribution verändern. Wenn eine Veröffentlichung unmittelbar bevorsteht, werden auch das Sicherheitsteam oder die Paketbetreuer direkt Aktualisierungen für Testing zur Verfügung stellen.

Zusätzlich kann auch das Debian-Sicherheits-Testing-Team (`http://secure-testing-master.debian.net`) Debian-Sicherheits-Testing-Ankündigungen (DTSA) für Pakete im Testing-Zweig herausgeben, wenn sofort eine Lücke in diesem Zweig geschlossen werden muss und die normale Vorgehensweise nicht abgewartet werden kann (oder die übliche Vorgehensweise durch andere Pakete blockiert ist).

Benutzer, die von diesem Angebot Gebrauch machen wollen, müssen folgende Zeilen ihrer `/etc/apt/sources.list` (anstatt der Zeilen, die unter 'Ausführen von Sicherheitsupdates' auf Seite 48 dargestellt wurden) hinzufügen:

```
deb http://secure-testing.debian.net/debian-secure-testing etch/security-u
# Diese Zeile macht es möglich, auch Quellpakete herunterzuladen
# FIXME: donwload
deb-src http://secure-testing.debian.net/debian-secure-testing etch/securi
```

Für weitere Informationen zu diesem Angebot können Sie die entsprechende Ankündigung (`http://lists.debian.org/debian-devel-announce/2005/09/msg00006.html`) lesen. Dieses Angebot startete offiziell im September 2005.

10.1.5 Automatische Aktualisierungen in einem Debian GNU/Linux System

Es sei vorweggeschickt, dass automatische Aktualisierungen nicht vollständig empfohlen werden, da Administratoren die DSAs durchsehen und die Bedeutung einer bestimmten Sicherheitsaktualisierung verstehen sollten.

Wenn Sie Ihr System automatisch aktualisieren wollen, sollten Sie Folgendes durchführen:

- Konfigurieren Sie `apt` so, dass Pakete, die Sie nicht aktualisieren wollen, ihrer momentane Version beibehalten. Das können Sie entweder mit einer Eigenschaft von `apt`, dem *pinning* (festheften), erreichen, oder Sie kennzeichnen sie mit `dpkg` oder `dselect` als *hold* (festgehalten).

  Um Pakete einer bestimmten Veröffentlichung mit pinning festzuheften, müssen Sie `/etc/apt/preferences` bearbeiten (siehe `apt_preferences(5)`) und Folgendes hinzufügen:

  ```
  Package: *
  Pin: release a=stable
  Pin-Priority: 100
  ```

FIXME: verify if this configuration is OK.

- Entweder setzen Sie `cron-apt` ein, wie in 'Automatisches Überprüfung von Aktualisierungen mit cron-apt' auf Seite 182 beschrieben wird, und erlauben ihm, heruntergeladene Pakete zu installieren. Oder Sie fügen selbst einen Eintrag für `cron` hinzu, damit die Aktualisierung täglich ausgeführt wird. Ein Beispiel:

  ```
  apt-get update && apt-get -y upgrade
  ```

  Die Option `-y` veranlasst `apt`, für alle Fragen, die während der Aktualisierung auftreten können, 'yes' anzunehmen. In manchen Fällen sollten Sie die Option `--trivial-only` (nur Bagatellen) der Option `--assume-yes` (ist gleichbedeutend mit `-y`) vorziehen.[1]

- Richten Sie `cron` so ein, dass `debconf` während der Aktualisierung keine Eingabe verlangt. Auf diese Weise können Aktualisierungen nicht-interaktiv durchgeführt werden.[2]

- Überprüfen Sie die Ergebnisse der Ausführung von `cron`, die an den Superuser gemailt werden (sofern nicht die Umgebungsvariable `MAILTO` im Skript geändert wurde).

Eine sichere Alternative könnte es sein, die Option `-d` (oder `--download-only`) zu verwenden. Das hat zur Folge, dass die benötigten Pakete nur heruntergeladen, aber nicht installiert werden. Und wenn dann die Ausführung von `cron` zeigt, dass das System aktualisiert werden muss, kann das von Hand vorgenommen werden.

Um diese Aufgaben zu erfüllen, muss das System korrekt konfiguriert sein, um Sicherheitsaktualisierungen herunterzuladen. Dies wurde in 'Ausführen von Sicherheitsupdates' auf Seite 48 diskutiert.

Allerdings wird dieses Vorgehen ohne eine genaue Analyse nicht für *Unstable* empfohlen, da Sie Ihr System in einen unbrauchbaren Zustand bringen können, wenn sich ein gravierender Fehler in ein wichtiges Paket eingeschlichen hat und auf Ihrem System installiert wird. *Testing* ist vor diesem Problem etwas besser *geschützt*, da gravierende Fehler eine bessere Chance haben entdeckt zu werden, bevor das Paket in den Testing-Zweig wandert (obwohl Ihnen trotzdem *keine* Sicherheitsaktualisierungen zur Verfügung stehen).

Wenn Sie eine gemischte Distribution haben, also eine Installation von *Stable* mit einige Pakete aus *Testing* oder *Unstable*, können Sie mit den Pinning-Eigenschaften oder der Option `--target-release` von `apt-get` herumspielen, um *nur* die Pakete zu aktualisieren, die Sie früher aktualisiert haben.[3]

[1] Sie können auch die Option `--quiet` (`-q`) verwenden. Sie verringert die Ausgabe von `apt-get` und wird keine Ausgabe produzieren, wenn keine Pakete installiert werden.

[2] Beachten Sie, dass einige Pakete *nicht* `debconf` verwenden könnten. Die Aktualisierung könnte dann hängen bleiben, da Pakete während ihrer Konfiguration Eingaben des Nutzers verlangen.

[3] Dies ist ein verbreitetes Problem, da viele Nutzer ein stabiles System betreiben wollen, aber einige Pakete aus *Unstable* einsetzen, um die neusten Funktionen zu haben. Das kommt daher, dass sich manche Projekte schneller entwickeln als die Veröffentlichungen von Debians *Stable*.

10.2 Periodische Überprüfung der Integrität

Mit Hilfe der Basisinformationen, die Sie nach der Installation erstellt haben (also mit dem Schnappschuss, der in 'Einen Schnappschuss des Systems erstellen' auf Seite 94 beschrieben wird, sollte es Ihnen möglich sein, von Zeit zu Zeit die Integrität des Systems zu überprüfen. Eine Integritätsprüfung kann Veränderungen am Dateisystem entdecken, die durch einen Eindringling oder einen Fehler des Systemadministrators entstanden sind.

Überprüfungen der Integrität sollen, wenn möglich, von außerhalb durchgeführt werden.[4] Das bedeutet, dass das Betriebssystem des überprüften Systems nicht verwendet wird, um den falschen Eindruck von Sicherheit (also falsche Negative) zu verhindern, der z.B. durch installierte Rootkits entstehen könnte. Die Datenbank, mit der das System verglichen wird, sollte sich daher auf einem nur-lesbaren Medium befinden.

Falls der Einsatz eines außenstehenden Systems keine Möglichkeit ist, sollten Sie in Betracht ziehen, die Integritätsprüfung mit den verfügbaren Werkzeugen zur Prüfung der Integrität des Dateisystem durchzuführen. Allerdings sollten Vorsichtsmaßnahmen getroffen werden: Die Datenbank für die Integritätsprüfung sollte nur-lesbar sein, und Sie sollten auch sicherstellen, dass das Programm, das die Integrität überprüft, (und der Kernel des Betriebssystems) nicht manipuliert wurde.

Einige Werkzeuge, die im Abschnitt über Programme zur Integritätsprüfung beschrieben wurden, wie z.B. `aide`, `integrit` und `samhain`, sind schon so eingerichtet, dass sie regelmäßige Nachprüfungen durchführen (mittels crontab in den ersten beiden Fällen und mittels eines eigenständigen Daemons bei `samhain`). Sie können den Administrator auf verschiedenen Wegen warnen (normalerweise E-Mail, aber `samhain` kann auch Seiten, SNMP-Traps oder einen Alarm an syslog schicken), wenn sich das Dateisystem verändert.

Wenn Sie eine Sicherheitsaktualisierung des System vorgenommen haben, müssen Sie natürlich den Schnappschuss des Systems neu aufzeichnen, um ihn an die Änderungen durch die Sicherheitsaktualisierung anzupassen.

10.3 Aufsetzen einer Eindringlingserkennung

Debian GNU/Linux enthält Programme zur Erkennung von Eindringlingen. Das sind Programme, die unpassende oder bösartige Aktivitäten auf Ihrem lokalen System oder auf anderen System in Ihrem lokalen Netzwerk entdecken. Diese Art von Verteidigung ist wichtig, wenn das System sehr entscheidend ist oder Sie wirklich unter Verfolgungswahn leiden. Die gebräuchlichsten Herangehensweisen sind die statistische Entdeckung von Unregelmäßigkeiten und die Entdeckung bestimmter Muster.

Beachten Sie immer, dass Sie einen Alarm-und-Antwort-Mechanismus brauchen, um Ihre Systemsicherheit mit einer dieser Werkzeuge wirklich zu verbessern. Eindringlingserkennung ist Zeitverschwendung, wenn Sie niemanden alarmieren werden.

[4] Ein leichter Weg, das ist tun, ist die Verwendung einer Live-CD wie Knoppix Std (`http://www.knoppix-std.org/`), die sowohl die Programme zur Integritätsprüfung als auch die dazugehörige Datenbank enthält.

Wenn ein bestimmter Angriff entdeckt worden ist, werden die meisten Programme zur Eindringlingserkennung entweder den Vorfall mit `syslog` protokollieren oder E-Mails an Root schicken (der Empfänger der E-Mails kann normalerweise eingestellt werden). Ein Administrator muss die Programme passend konfigurieren, so dass falsche Positivmeldungen keinen Alarm auslösen. Alarme können auf einen laufenden Angriff hindeuten und wären später – sagen wir mal am nächsten Tag – nicht mehr nützlich, da der Angriff dann bereits erfolgreich beendet worden sein könnte. Stellen Sie also sicher, dass es eine passende Regelung über die Handhabung von Alarmen gibt, und dass technische Maßnahmen zur Umsetzung dieser Regelung vorhanden sind.

Eine interessante Quelle für Information ist CERT's Intrusion Detection Checklist (`http://www.cert.org/tech_tips/intruder_detection_checklist.html`)

10.3.1 Netzwerk basierende Eindringlingserkennung

Programme, die der Netzwerk basierende Eindringlingserkennung dienen, überwachen den Verkehr eines Netzwerkabschnitts und arbeiten auf Grundlage dieser Daten. Genauer ausgedrückt, es werden die Pakete im Netzwerk untersucht, um festzustellen, ob sie mit bestimmten Merkmalen übereinstimmen.

`snort` ist ein vielseitiger Paketschnüffler oder -logger, der Angriffe mit einer Bibliothek von Angriffssignaturen erkennt. Es erkennt eine breite Palette von Angriffen und Tests, wie zum Beispiel Pufferüberläufe, verdecktes Abtasten von Ports (stealth port scans), CGI Angriffe, SMB Tests und vieles mehr. `snort` hat auch die Fähigkeit, einen zeitnahen Alarm auszulösen. Dies ist in Werkzeug, das auf jedem Router installiert werden sollte, um ein Auge auf Ihr Netzwerk zu haben. Installieren Sie es einfach mit `apt-get install snort`, beantworten Sie den Fragen und beobachten Sie die logs. Für einen etwas breiteren Sicherheitsrahmen sollten Sie sich Prelude (`http://www.prelude-ids.org`) ansehen.

Debians Paket `Snort` hat viele Sicherheitstests standardmäßig eingeschaltet. Jedoch sollten Sie die Konfiguration anpassen, um die Dienste, die auf Ihrem System laufen, zu berücksichtigen. Sie möchten vielleicht auch zusätzliche Tests speziell für diese Dienste nutzen.

Achtung: Die Snort-Pakete aus Woody sind ziemlich veraltet und könnten sogar fehlerhaft (`http://lists.debian.org/debian-devel/2003/debian-devel-200308/msg02105.html`) sein. Sie erhalten zurückportierte (und signierte) Snort-Pakete, die vom Paketbetreuer angeboten werden, unter `http://people.debian.org/~ssmeenk/snort-stable-i386/`.

Es gibt noch andere, einfachere Werkzeuge, die dazu benutzt werden können, Angriffe auf das Netzwerk zu erkennen. `Portsentry` ist ein interessantes Paket, dass Sie warnen kann, wenn jemand Ihre Rechner scannt. Auch andere Programme wie `ippl` oder `iplogger` erkennen bestimmte IP (TCP und ICMP) Angriffe, auch wenn sie nicht so fortgeschrittenen Techniken zur Erkennung von Netzwerkangriffen wie `snort` haben.

Sie können jedes dieser Werkzeuge mit dem Paket `idswakeup` testen. Das ist ein Shell-Skript, das falsche Alarme verursacht und Signaturen vieler gebräuchlicher Angriffe enthält.

10.3.2 Host basierende Eindringlingserkennung

Eine Eindringlingserkennung, die auf einem Host basiert, beruht darauf, Software auf dem zu überwachenden System zu laden, die Log-Dateien und die Auditing-Programme des Systems als Datengrundlage verwendet. Sie sucht nach verdächtigen Prozessen, kontrolliert den Zugang zum Host und überwacht u.U. auch Änderungen an kritischen Systemdateien.

`tiger` ist ein älteres Programm zur Eindringlingserkennung, dass seit der Woody-Distribution auf Debian portiert wurde. `tiger` bietet Tests von verbreiteten Problemen in Zusammenhang mit Einbrüchen, wie der Stärke von Passwörtern, Problemen mit dem Dateisystem, kommunizierenden Prozessen und anderen Möglichkeiten, mit denen Root kompromittiert werden könnte. Dieses Paket umfasst neue, debianspezifische Sicherheitstests, einschließlich der MD5-Summen von installierten Programmen, des Orts von Dateien, die zu keinem Paket gehören und einer Analyse von lokalen, lauschenden Prozessen. Die Standardinstallation lässt `tiger` einmal am Tag laufen und einen Bericht erstellen, der an den Superuser geschickt wird und Informationen zu möglichen Kompromittierungen enthält.

Programme zur Protokollanalyse, wie zum Beispiel `logcheck` können zusätzliche benutzt werden, um Einbruchsversuche zu erkennen. Siehe 'Nutzung und Anpassung von `logcheck`' auf Seite 75.

Daneben können Pakete, die die Integrität des Dateisystems überwachen (siehe 'Prüfung der Integrität des Dateisystems' auf Seite 85), sehr nützlich sein, um Anomalien in einer abgesicherten Umgebung zu erkennen. Ein erfolgreicher Einbruch wird höchstwahrscheinlich Dateien auf dem lokalen Dateisystem verändern, um die lokalen Sicherheitsregelungen zu umgehen, Trojaner zu installieren oder Nutzer zu erstellen. Solche Ereignisse können mit Prüfwerkzeugen der Dateisystemintegrität erkannt werden.

10.4 Vermeiden von Root-Kits

10.4.1 Ladbare Kernel-Module (LKM)

Ladbare Kernel-Module sind Dateien, die nachladbare Teile des Kernels enthalten. Sie werden dazu verwendet, die Funktionalität des Kernel zu erweitern. Der Hauptnutzen des Einsatzes von Modulen liegt darin, dass Sie zusätzliche Geräte wie eine Ethernet- oder Soundkarte hinzuzufügen können, ohne dass die Kernelquelle gepatcht und der gesamte Kernel neu übersetzt werden müsste. Allerdings können Cracker LKMs für Root-Kits (knark und adore) benutzen, um auf GNU/Linux Systemen Hintertüren zu öffnen.

LKM-Hintertüren sind ausgeklügelter und schwere zu entdecken als traditionelle Root-Kits. Sie können Prozesse, Dateien, Verzeichnisse und sogar Verbindungen verstecken, ohne den Quellcode der Programme verändern zu müssen. Zum Beispiel kann ein bösartiges LKM den Kernel dazu zwingen, bestimmte Prozesses vor `procfs` zu verstecken, so dass nicht einmal eine bekanntermaßen gute Kopie des Programms `ps` alle Informationen über die aktuellen Prozesse korrekt auflisten.

10.4.2 Erkennen von Root-Kits

Es gibt zwei Herangehensweisen, um Ihr System gegen LKM-Root-Kits zu verteidigen: die aktive Verteidigung und die reaktive Verteidigung. Die Sucharbeit kann einfach und schmerzlos sein, oder schwierig und ermüdend, ganz abhängig von der Maßnahme, die Sie ergreifen.

Aktive Verteidigung

Der Vorteil dieser Art der Verteidigung ist, dass schon verhindert wird, dass das System Schaden nimmt. Eine mögliche Strategie ist *das Ziel zuerst zu erreichen*, also ein LKM zu laden, das dazu da ist, das System vor anderen böswilligen LKMs zu schützen. Eine andere Maßnahme ist es, dem Kernel Fähigkeiten zu entziehen. Zum Beispiel können Sie aus dem Kernel vollständig die Fähigkeit von ladbaren Kernel-Modulen entfernen. Beachten Sie allerdings, dass es Root-Kits gibt, die selbst in diesen Fällen funktionieren. Es gibt auch welche, die direkt `/dev/kmem` (Kernelspeicher) manipulieren, um sich zu verstecken.

Debian GNU/Linux hat ein paar Pakete, die dazu verwendet werden können, eine aktive Verteidigung aufzusetzen:

- `lcap` - eine benutzerfreundliche Schnittstelle, um dem Kernel *Fähigkeiten* zu entziehen (kernelbasierte Zugriffskontrolle), um das System sicherer zu machen. Beispielsweise wird das Ausführen von `lcap CAP_SYS_MODULE` [5] die Fähigkeit der ladbaren Module entfernen (sogar für Root).[6] Weitere (etwas ältere) Informationen zu Kernelfähigkeiten finden Sie in Jon Corbets Abschnitt Kernel development (`http://lwn.net/1999/1202/kernel.php3`) auf LWN vom Dezember 1999.

Wenn Sie diese vielen Möglichkeiten auf Ihrem GNU/Linux System nicht wirklich brauchen, sollten Sie die Unterstützung für ladbare Module während der Konfiguration des Kernels abschalten. Das erreichen Sie, indem Sie einfach CONFIG_MODULES=n während des Konfiguration Ihres Kernels oder in der Datei `.config` festsetzen. So werden LKM Root-Kits vermieden, aber Sie verlieren eine leistungsfähige Eigenschaft des Linux-Kernels. Außerdem kann das Abschalten der nachladbaren Module den Kernel überladen, so dass die Unterstützung ladbarer Module notwendig wird.

Reaktive Verteidigung

Der Vorteil reaktiver Verteidigung ist, dass sie die Systemressourcen nicht überlädt. Sie funktioniert durch das Vergleichen von einer Tabelle der Systemaufrufe mit einer bekanntermaßen

[5] Es gibt über 28 Fähigkeiten einschließlich `CAP_BSET`, `CAP_CHOWN`, `CAP_FOWNER`, `CAP_FSETID`, `CAP_FS_MASK`, `CAP_FULL_SET`, `CAP_INIT_EFF_SET`, `CAP_INIT_INH_SET`, `CAP_IPC_LOCK`, `CAP_IPC_OWNER`, `CAP_KILL`, `CAP_LEASE`, `CAP_LINUX_IMMUTABLE`, `CAP_MKNOD`, `CAP_NET_ADMIN`, `CAP_NET_BIND_SERVICE`, `CAP_NET_RAW`, `CAP_SETGID`, `CAP_SETPCAP`, `CAP_SETUID`, `CAP_SYS_ADMIN`, `CAP_SYS_BOOT`, `CAP_SYS_CHROOT`, `CAP_SYS_MODULE`, `CAP_SYS_NICE`, `CAP_SYS_PACCT`, `CAP_SYS_PTRACE`, `CAP_SYS_RAWIO`, `CAP_SYS_RESOURCE`, `CAP_SYS_TIME` und `CAP_SYS_TTY_CONFIG`. Alle können deaktiviert werden, um Ihren Kernel abzuhärten.

[6] Um dies tun zu können, müssen Sie nicht `lcap` installieren, aber damit ist es einfacher, als von Hand `/proc/sys/kernel/cap-bound` anzupassen.

sauberen Kopie (`System.map`). Eine reaktive Verteidigung kann den Systemadministrator natürlich nur benachrichtigen, wenn das System bereits kompromittiert wurde.

Die Entdeckung von Root-Kits vollbringt unter Debian `chkrootkit`. Das Programm Chkrootkit (`http://www.chkrootkit.org`) prüft Anzeichen von bekannten Root-Kits auf dem Zielsystem. Es ist aber kein völlig sicherer Test.

10.5 Geniale/paranoide Ideen — was Sie tun können

Dies ist wahrscheinlich der unsicherste und lustigste Abschnitt, da ich hoffe, dass manche der "Wow, das klingt verrückt"-Ideen umgesetzt werden. Im folgenden werden nur ein paar Ideen vorgestellt, wie Sie Ihre Sicherheit erhöhen können — abhängig von Ihrem Standpunkt aus können Sie sie für genial, paranoid, verrückt oder sicher halten,

- Mit Pluggable Authentication Modules (PAM) herum spielen. Wie in einem phrack 56 Artikel geschrieben wurde, ist das schöne an PAM, dass "Ihrer Fantasie keine Grenzen gesetzt sind." Das stimmt. Stellen Sie sich vor, Root kann sich nur mit einen Fingerabdruck oder Abtastung des Auges oder einer Kryptokarte einloggen (warum habe ich hier nur "oder" und nicht "und" gesagt?).
- Faschistisches Logging. Ich würde sagen, dass alles, was wir bisher über Logging besprochen haben, unter "weiches Loggen" fällt. Wenn Sie echtes Logging betreiben wollen, besorgen Sie sich einen Drucker mit Endlos-Papier und schicken ihm alle Logs. Hört sich lustig an, ist aber zuverlässig und kann nicht manipuliert oder entfernt werden.
- CD-Distribution. Diese Idee ist sehr leicht zu realisieren und bewirkt ganz gute Sicherheit. Erstellen Sie eine abgesicherte Debian Distribution mit passenden Firewall-Regeln. Erstellen Sie davon ein bootbares ISO-Image und brennen Sie es auf eine CD-ROM. Jetzt haben Sie eine nur lesbare Distribution mit etwa 600 MB Speicherplatz für Dienste. Stellen Sie lediglich sicher, dass alle Daten, die geschrieben werden sollen, übers Netz geschrieben werden. Für einem Eindringling ist es unmöglich, Schreibzugriff auf diesem System zu erhalten. Alle Änderungen, die ein Eindringling vornimmt, werden mit einem Reboot des Systems rückgängig gemacht.
- Schalten Sie die Modul-Fähigkeiten des Kernels ab. Wenn Sie die Nutzung von Kernel-Modulen während der Kernel-Kompilierung abschalten, werden viele Kernel basierende Hintertüren nicht einsetzbar, da die meisten von ihnen darauf basieren, modifizierte Kernel-Module zu installieren (siehe oben).
- Loggen über ein serielles Kabel (von Gaby Schilders). So lange Server immer noch serielle Schnittstellen haben: Stellen Sie sich vor, Sie ein Log-System für eine Anzahl von Servern. Es ist vom Netz abgeschnitten und mit den Servern über einen Multiplexer für serielle Schnittstellen (Cyclades oder ähnliches) verbunden. Jetzt sollen alle Ihre Server über ihre serielle Schnittstelle loggen. Einfach nur hinschreiben. Die Log-Maschine akzeptiert nur einfachen Text als Eingabe auf ihrer seriellen Schnittstelle und schreibt ihn lediglich in eine Log-Datei. Schließen Sie einen CD- oder DVD-Brenner an. Brennen Sie

die Log-Datei, wenn sie die Größe des Mediums erreicht hat. Wenn es jetzt nur noch CD-Brenner mit automatischem Medien-Wechsel gäbe ... Nicht so dauerhaft gespeichert wie ein Ausdruck, aber mit dieser Methode kann man größere Mengen handhaben, und die CD-ROMs nehmen nicht so viel Platz weg.

- Ändern Sie die Dateiattribute mit `chattr` (dem Tipps-HOWTO von Jim Dennis entnommen). Nachdem Sie Ihr System sauber installiert und konfiguriert haben, verwenden Sie das Programm `chattr` mit dem Attribut `+i`, um Dateien unveränderbar zu machen (die Datei kann nicht gelöscht, umbenannt, verlinkt oder beschrieben werden). Sie sollten dieses Attribut für alle Dateien in `/bin`, `/sbin/`, `/usr/bin`, `/usr/sbin`, `/usr/lib` und den Kerneldateien in root. Sie können auch eine Kopie aller Dateien in `/etc/` mit `tar` oder dergleichen erstellen und das Archiv als unveränderbar kennzeichnen.

  Mit dieser Vorgehensweise können Sie den Schaden zu begrenzen, den Sie anrichten können, wenn Sie als Root eingeloggt sind. Sie können keine Dateien mit einer fehlgeleiteten Umleitung überschreiben, und Sie werden Ihr System nicht durch ein fehlplatziertes Leerzeichen im Kommando `rm -fr` unbenutzbar machen (Sie können aber Ihren Daten immer noch einigen Schaden zufügen — aber Ihre Bibliotheken und Programme sind sicherer).

  Dies macht auch verschiedene Sicherheits- und Denial-of-Service (DoS) Exploits entweder unmöglich oder weitaus schwieriger (da viele von ihnen darauf beruhen, Dateien durch Aktionen eines SETUID-Programms zu überschreiben, das *keinen frei wählbaren Shellbefehl zur Verfügung stellt*.

  Eine Unbequemlichkeit dieser Vorgehensweise macht sich bemerkbar, wenn Sie verschiedene Systemprogramme bauen und installieren. Auf der anderen Seite verhindert dies auch, dass `make install` die Dateien überschreibt. Wenn Sie vergessen, das Makefile zu lesen, und die Dateien, die überschrieben werden sollen, mit chattr -i behandelt haben (und die Verzeichnisse, in denen Sie neue Dateien erstellen wollen), schlägt der make-Befehl fehl. Sie müssen nur das Kommando `chattr` ausführen und make neu aufrufen. Sie können diese Gelegenheit gleich dazu benutzen, Ihre alten bin's und libs auszumisten und sie z.B. in ein .old/-Verzeichnis oder Tar-Archiv zu verschieben.

  Beachten Sie, dass dies Sie auch daran hindert, die Pakete Ihres Systems zu aktualisieren, da die Dateien aus den Paketen nicht überschrieben werden können. Also sollten Sie vielleicht ein Skript oder einen anderen Mechanismus haben, der das immutable-Flag auf allen Dateien deaktiviert, bevor Sie ein `apt-get update` ausführen.

  Spielen Sie mit der UTP-Verkabelung herum. Schneiden Sie dazu zwei oder vier Kabel durch und stellen ein Kabel her, das nur Verkehr in eine Richtung zulässt. Verwenden Sie dann UDP-Pakete, um Informationen an die Zielmaschine zu schicken, die ein sicherer Log-Server oder ein System zur Speicherung von Kreditkartennummern sein kann.

10.5.1 Aufstellen eines Honigtopfes (honeypot)

Ein Honigtopf ist ein System, das darauf ausgelegt ist, Systemadministratoren beizubringen, wie Cracker ein System abtasten und darin einbrechen. Es ist eine Systemeinstellung mit der Erwartung und dem Zweck, dass das System abgetastet und angegriffen und möglicherweise

darin eingebrochen wird. Wenn Systemadministratoren erfahren, welche Werkzeuge und Methoden Cracker anwenden, können sie daraus lernen, wie sie ihr System und Netzwerk besser schützen.

Debian GNU/Linux-Systeme können leicht als Honigtopf eingerichtet werden, wenn Sie Zeit opfern, sie aufzusetzen und zu überwachen. Sie können leicht den gefälschten Server, die Firewall[7], die den Honigtopf überwacht, und ein Programm, das Eindringling ins Netzwerk entdecken kann, einrichten. Verbinden Sie den Honigtopf mit dem Internet und warten Sie ab. Stellen Sie sicher, dass Sie rechtzeitig alarmiert werden (siehe 'Die Wichtigkeit von Logs und Alarmen' auf Seite 74), wenn in das System eingedrungen wird, damit Sie geeignete Schritte einleiten und den Angriff beenden können, wenn Sie genug gesehen haben. Hier folgen einige Pakete und Probleme, die Sie in Betracht ziehen sollten, wenn Sie einen Honigtopf einrichten:

- Die Firewall-Technologie, die Sie verwenden (verfügbar durch den Linux-Kernel).
- `syslog-ng`. Nützlich, um Logs des Honigtopfs zu einem entfernen Syslog-Server zu schicken.
- `snort`, um allen eingehenden Netzwerkverkehr auf den Honigtopf mitzuschneiden und die Angriffe zu erkennen.
- `osh`, eine eingeschränkte Shell mit Logging, die unter SETUID-Root läuft und verbesserte Sicherheit hat (siehe den Artikel von Lance Spitzner weiter unten).
- Natürlich alle Daemons, die Sie auf dem falschen Honigtopfserver verwenden wollen. Je nachdem, welche Art von Angreifer Sie analysieren wollen, können Sie den Honigtopf abhärten und die Sicherheitsaktualisierungen einspielen (oder eben *nicht*).
- Integritätsprüfer (siehe 'Prüfung der Integrität des Dateisystems' auf Seite 85) und das Coroner's Toolkit (`tct`), um nach dem Angriff eine Analyse durchzuführen.
- `honeyd` und `farpd`, um einen Honigtopf einzurichten, der auf Verbindungen zu ungenutzten IP-Adressen lauscht und diese an Skripte weiterleitet, die echte Dienste simulieren. Sehen Sie sich auch `iisemulator` an.
- `tinyhoneypot`, um einen einfachen Honigtopf-Server mit gefälschten Diensten einzurichten.

Falls Sie kein System übrig haben, um die Honigtöpfe und Systeme, die das Netzwerk schützen und kontrollieren, zu bauen, können Sie die Technologie zur Virtualisierung einsetzen, die in `xen` oder `uml` (User-Mode-Linux) enthalten ist. Wenn Sie diesen Weg wählen, müssen Sie Ihren Kernel entweder mit `kernel-patch-xen` oder `kernel-patch-uml` patchen.

Sie können mehr über das Aufstellen eines Honigtopfs in Lanze Spitzners exzellentem Artikel To Build a Honeypot (`http://www.net-security.org/text/articles/spitzner/honeypot.shtml`) (aus der Know your Enemy Serie). Außerdem stellt das Honeynet Project (`http://project.honeynet.org/`) wertvolle Informationen über das Aufstellen von Honigtöpfen und der Analyse von Angriffen auf sie zur Verfügung.

[7] Sie sollten typischerweise eine Bridge-Firewall einsetzen, damit die Firewall selbst nicht entdeckt werden kann. Lesen Sie mehr dazu unter 'Aufsetzen einer überbrückenden Firewall (bridge Firewall)' auf Seite 237.

Kapitel 11

Nach einer Kompromittierung (Reaktion auf einem Vorfall)

11.1 Allgemeines Verhalten

Wenn Sie während eines Angriffs physisch anwesend sind, sollte Ihre erste Reaktion sein, den Rechner vom Netzwerk zu trennen, indem Sie das Kabel aus der Netzwerkkarte ziehen (wenn das keinen nachteiligen Einfluss auf Ihre Geschäfte hat). Das Netzwerk auf Schicht 1 abzuschalten ist der einzig wirklich erfolgreiche Weg, um den Angreifer aus dem gehackten Rechner herauszuhalten (weiser Ratschlag von Phillip Hofmeister).

Allerdings können einige Werkzeuge, die durch Rootkits, Trojaner oder sogar unehrlichen Benutzern über eine Hintertür installiert wurden, diesen Vorgang erkennen und auf ihn reagieren. Es ist nicht wirklich lustig, wenn Sie sehen, dass `rm -rf /` ausgeführt wird, wenn Sie das Netzwerkkabel ziehen. Wenn Sie nicht bereit sind, dieses Risiko einzugehen, und Sie sich sicher sind, dass in das System eingebrochen wurde, sollten Sie *das Stromkabel herausziehen* (alle, wenn es mehr als eines gibt) und Ihre Daumen drücken. Das hört sich zwar extrem an, verhindert aber tatsächlich eine Logikbombe, die ein Eindringling programmiert haben könnte. Auf jeden Fall sollte ein kompromittiertes System *nicht neugestartet* werden. Entweder sollten die Festplatten in einem anderen System analysiert werden, oder Sie sollten ein anderen Medium (eine CD-ROM) benutzen, um das System zu booten und analysieren. Sie sollten *nicht* die Rettungsdisk von Debian verwenden, um das System zu starten. Sie *können* aber die Shell auf der Installationsdisk benutzen (wie Sie wissen, erreichen Sie sie mit Alt+F2), um das System zu analysieren. [1]

Die beste Methode, um ein gehacktes System wiederherzustellen, ist, ein Live-Dateisystem auf einer CD-ROM mit allen Programmen (und Kernel-Modulen) verwenden, die Sie brauchen, um auf das eingebrochene System zugreifen zu können. Sie können das Paket `mkinitrd-cd`

[1] Wenn Sie abenteuerlustig sind, sollten Sie sich am System anmelden und die Informationen aller laufenden Prozesse speichern (Sie bekommen eine Menge aus /proc/nnn/). Es ist möglich, den gesamten ausführbaren Code aus dem Arbeitsspeicher zu ziehen, sogar dann, wenn der Angreifer die ausführbaren Dateien von der Festplatte gelöscht hat. Ziehen Sie danach das Stromkabel.

benutzen, um eine solche CD-ROM zu erstellen [2]. Auch die CD-ROM von FIRE (http://biatchux.dmzs.com/) (früher als Biatchux bekannt) könnte hilfreich sein, da diese Live-CD-ROM forensische Werkzeuge enthält, die in solchen Situationen nützlich sind. Es gibt (noch) kein Programm wie dieses, das auf Debian basiert. Es gibt auch keinen leichten Weg, eine CD-ROM mit Ihrer Auswahl von Debian-Paketen und `mkinitrd-cd` zu erstellen. Daher werden Sie die Dokumentation lesen müssen, wie Sie Ihre eigenen CD-ROMs machen.

Wenn Sie eine Kompromittierung wirklich schnell reparieren wollen, sollten Sie den kompromittierten Rechner aus dem Netzwerk entfernen und das Betriebssystem von Grund auf neu installieren. Dies könnte natürlich nicht sehr wirkungsvoll sein, da Sie nicht erfahren, wie der Eindringling zuvor Root-Rechte bekommen hat. Um das herauszufinden, müssen Sie alles prüfen: Firewall, Integrität der Dateien, Log-Host, Log-Dateien und so weiter. Weitere Informationen, was Sie nach einem Einbruch unternehmen sollten, finden Sie unter Sans' Incident Handling Guide (http://www.sans.org/y2k/DDoS.htm) oder CERT's Steps for Recovering from a UNIX or NT System Compromise (http://www.cert.org/tech_tips/root_compromise.html).

Einige häufige Fragen, wie mit einem gehackten Debian-GNU/Linux-System umzugehen ist, sind unter 'Mein System ist angreifbar! (Sind Sie sich sicher?)' auf Seite 213 zu finden.

11.2 Anlegen von Sicherheitskopien Ihres Systems

wenn Sie sich sicher sind, dass das System kompromittiert wurde, vergessen Sie nicht, dass Sie weder der installierten Software noch irgendwelchen Informationen, die es an Sie liefert, vertrauen können. Anwendungen könnten von einem Trojaner befallen sein, Kernel-Module könnten installiert worden sein, usw.

Am besten ist es, eine komplette Sicherheitskopie Ihres Dateisystems (mittels `dd`) zu erstellen, nachdem Sie von einem sicheren Medium gebootet haben. Debian GNU/Linux CD-ROMs können dazu nützlich sein, da sie auf Konsole zwei eine Shell anbieten, nachdem die Installation gestartet wurde (mit Alt+2 und Enter aktivieren Sie sie). Von dieser Shell aus sollten Sie eine Sicherheitskopie möglichst auf einem anderen Host erstellen (vielleicht auf einen Netzwerk-File-Server über NFS/FTP). Dadurch kann eine Analyse des Einbruchs oder eine Neuinstallation durchgeführt werden, während das betroffene System offline ist.

Wenn Sie sich sicher sind, dass es sich lediglich um ein trojanisiertes Kernel-Modul handelt, können Sie versuchen, das Kernel-Image von der Debian-CD-ROM im *rescue*-Modus zu laden. Stellen Sie sicher, dass Sie im *single*-Modus starten, so dass nach dem Kernel keine weiteren Trojaner-Prozesse gestartet werden.

[2] Das ist auch das Werkzeug, mit dem die CD-ROMs für das Projekt Gibraltar (http://www.gibraltar.at/) erstellt werden. Das ist eine Firewall auf einer Live-CD-ROM, die auf der Debian-Distribution beruht.

11.3 Setzen Sie sich mit dem lokal CERT in Verbindung

Das CERT (Computer and Emergency Response Team) ist eine Organisation, die Ihnen helfen kann, Ihr System nach einem Einbruch wiederherzustellen. Es gibt CERTs weltweit [3]. Sie sollten mit dem lokalen CERT Verbindung aufnehmen, wenn sich ein sicherheitsrelevanter Vorfall ereignet hat, der zu einem Einbruch in Ihr System geführt hat. Die Menschen in der lokalen CERT können Ihnen helfen, Ihr System wiederherzustellen.

Selbst wenn Sie keine Hilfe benötigen, kann es anderen helfen, wenn Sie dem lokalen CERT (oder dem Koordinationszentrum des CERTs) Informationen des Einbruchs zur Verfügung stellen. Die gesammelten Informationen von gemeldeten Vorfällen werden verwendet, um herauszufinden, ob eine bestimmte Verwundbarkeit weit verbreitet ist, ob sich ein neuer Wurm ausbreitet oder welche neuen Angriffswerkzeuge eingesetzt werden. Diese Informationen werden benutzt, um die Internet-Gemeinschaft mit Informationen über die aktuellen Sicherheitsvorkommnisse (`http://www.cert.org/current/`) zu versorgen und um Hinweise zu Vorfällen (`http://www.cert.org/incident_notes/`) und sogar Anweisungen (`http://www.cert.org/advisories/`) zu veröffentlichen. Ausführliche Informationen, wie (und warum) ein Vorfall gemeldet wird, können Sie auf CERT's Incident Reporting Guidelines (`http://www.cert.org/tech_tips/incident_reporting.html`) nachlesen.

Sie können auch weniger formale Einrichtungen verwenden, wenn Sie Hilfe brauchen, um Ihr System wiederherzustellen, oder wenn Sie Informationen des Vorfalls diskutieren wollen. Dazu zählen die Mailingliste für Vorfälle (`http://marc.theaimsgroup.com/?l=incidents`) und die Mailingliste für Einbrüche (`http://marc.theaimsgroup.com/?l=intrusions`).

11.4 Forensische Analyse

Wenn Sie mehr Informationen sammeln wollen, enthält das Paket `tct` (The Coroner's Toolkit von Dan Farmer und Wietse Venema) Werkzeuge für eine *post mortem*-Analyse des Systems. `tct` erlaubt es dem Benutzer, Informationen über gelöschte Dateien, laufende Prozesse und mehr zu sammeln. Sehen Sie bitte für weitere Informationen in die mitgelieferte Dokumentation. Diese und andere Werkzeuge können auch auf Sleuthkit and Autopsy (`http://www.sleuthkit.org/`) von Brian Carrier, welches ein Web-Frontend zur forensischen Analyse von Disk-Images zur Verfügung stellt, gefunden werden. In Debian befindet sich sowohl `sleuthkit` (die Werkzeuge) und `autopsy` (die grafische Oberfläche).

[3] Dies ist eine Liste einiger CERTs. Ein vollständige Liste erhalten Sie unter FIRST Member Team information (`http://www.first.org/about/organization/teams/index.html`) (FIRST ist das Forum von Incident Response and Security Teams): AusCERT (`http://www.auscert.org.au`) (Australien), UNAM-CERT (`http://www.unam-cert.unam.mx/`) (Mexiko) CERT-Funet (`http://www.cert.funet.fi`) (Finnland), DFN-CERT (`http://www.dfn-cert.de`) (Deutschland), RUS-CERT (`http://cert.uni-stuttgart.de/`) (Deutschland), CERT-IT (`http://security.dico.unimi.it/`) (Italien), JPCERT/CC (`http://www.jpcert.or.jp/`) (Japan), UNINETT CERT (`http://cert.uninett.no`) (Norwegen), HR-CERT (`http://www.cert.hr`) (Kroatien) CERT Polskay (`http://www.cert.pl`) (Polen), RU-CERT (`http://www.cert.ru`) (Russland), SI-CERT (`http://www.arnes.si/si-cert/`) (Slowenien) IRIS-CERT (`http://www.rediris.es/cert/`) (Spanien), SWITCH-CERT (`http://www.switch.ch/cert/`) (Schweiz), TWCERT/CC (`http://www.cert.org.tw`) (Taiwan), und CERT/CC (`http://www.cert.org`) (USA).

Einige andere Programme aus der Debian-Distribution, die für forensische Analyse verwendet werden können, sind:

- `fenris`.
- `strace`.
- `ltrace`.

Alle diese Pakete können dazu benutzt werden, um Schurkenprogramme (wie z.B. Hintertüren) zu analysieren, um herauszufinden, wie sie arbeiten und was sie mit dem System anstellen. Einige andere gebräuchliche Werkzeuge sind `ldd` (in `libc6`), `strings` und `objdump` (beide in `binutils`).

Wenn Sie eine forensische Analyse mit Hintertüren oder verdächtigen Programmen durchführen, die Sie vom gehackten System haben, sollten Sie dies in einer sicheren Umgebung durchführen, z.B. in einem `bochs`-, oder `xen`-Image oder in einer `chroot`-Umgebung eines Nutzers mit geringen Rechten. Andernfalls könnte auch auf Ihrem eigenen System eine Hintertür eingerichtet oder R00t-Rechte erlangt werden.

Forensische Analysen sollten immer auf einer Sicherheitskopie der Daten angewendet werden, *niemals* auf die Daten selbst, da sie durch diese Analyse beeinflusst werden könnten und so Beweismittel zerstört werden würden.

Weiterführende Informationen über forensische Analyse können Sie in Dan Farmers und Wietse Venemas Buch Forensic Discovery (`http://www.porcupine.org/forensics/forensic-discovery/`) (online verfügbar), in ihrer Computer Forensics Column (`http://www.porcupine.org/forensics/column.html`) und in ihrem Computer Forensic Analysis Class Handouts (`http://www.porcupine.org/forensics/handouts.html`) finden. Eine weitere, sehr gute Quelle für Tipps zur forensischen Analyse ist Brian Carriers Newsletter The Sleuth Kit Informer (`http://www.sleuthkit.org/informer/index.php`). Auch die Honeynet Challenges (`http://www.honeynet.org/misc/chall.html`) sind eine ausgezeichnete Möglichkeit, Ihre forensischen Fähigkeiten zu verbessern, da sie echte Angriffe auf Honigtopfsysteme umfassen und Herausforderungen bieten, die von der forensischen Analyse von Festplatten bis zu Protokollen der Firewall und Paketerfassung alles beinhalten.

FIXME: This paragraph will hopefully provide more information about forensics in a Debian system in the coming future.

FIXME: Talk on how to do a debsums on a stable system with the MD5sums on CD and with the recovered file system restored on a separate partition.

FIXME: Add pointers to forensic analysis papers (like the Honeynet's reverse challenge or David Dittrich's papers (`http://staff.washington.edu/dittrich/`)).

Kapitel 12

Häufig gestellte Fragen / Frequently asked Questions (FAQ)

Dieses Kapitel führt Sie in ein paar der am häufigsten gestellten Fragen in der Security-Mailingliste von Debian ein. Sie sollten sie lesen, bevor Sie dort etwas posten, oder die Leute werden Ihnen "RTFM!" sagen.

12.1 Sicherheit im Debian Betriebssystem

12.1.1 Ist Debian sicherer als X?

Ein System ist so sicher, wie der Administrator fähig ist, es sicher zu machen. Debians Standardinstallation von Diensten zielt darauf ab, *sicher* zu sein. Sie ist aber nicht so paranoid wie andere Betriebssysteme, die Dienste *standardmäßig abgeschaltet*. In jedem Fall muss der Systemadministrator die Sicherheit des System den lokalen Sicherheitsmaßstäben anpassen.

Für eine Übersicht der Sicherheitslücken von vielen Betriebssystemen sollten Sie sich die US-CERT-Statistik (`http://www.cert.org/stats/cert_stats.html`) ansehen oder sich selber Statistiken mit der National Vulnerability Database (`http://nvd.nist.gov/statistics.cfm`) (früher ICAT) erstellen. Sind diese Daten nützlich? Es müssen verschiedene Faktoren berücksichtigt werden, wenn die Daten interpretiert werden sollen. Man sollte beachten, dass diese Daten nicht dazu verwendet werden können, um die Verwundbarkeit eines Betriebssystems mit der eines anderen zu vergleichen. [1] Bedenken Sie außerdem, dass sich einige registrierte Sicherheitslücken im Zusammenhang mit Debian nur auf den *Unstable*-Zweig, also den nicht offiziell veröffentlichten Zweig, beziehen.

[1] Zum Beispiel könnte es auf Grundlage einiger Daten scheinen, dass Windows NT sicherer ist als Linux. Dies wäre eine fragwürdige Annahme. Das liegt daran, dass Linux-Distributionen normalerweise viel mehr Anwendungen zur Verfügung stellen als Microsofts Windows NT. Dieses Problem des *Abzählens von Sicherheitslücken* wird besser in Why Open Source Software / Free Software (OSS/FS)? Look at the Numbers! (`http://www.dwheeler.com/oss_fs_why.html#security`) von David A. Wheeler beschrieben.

Ist Debian sicherer als andere Linux-Distributionen (wie Red Hat, SuSE, ...)?

Der Unterschied zwischen den Linux-Distributionen ist nicht sehr groß mit Ausnahme der Basisinstallation und der Paketverwaltung. Die meisten Distributionen beinhalten zum Großteil die gleichen Anwendungen. Der Hauptunterschied besteht in den Versionen dieser Programme, die mit der stabilen Veröffentlichung der Distribution ausgeliefert werden. Zum Beispiel sind der Kernel, Bind, Apache, OpenSSH, XFree, gcc, zlib, etc. in allen Linux-Distributionen vorhanden.

Ein Beispiel: Red Hat hatte Pech und wurde veröffentlicht, als foo 1.2.3 aktuell war. Später wurde darin eine Sicherheitslücke entdeckt. Dagegen hatte Debian das Glück, dass es mit foo 1.2.4 ausgeliefert wurde, in dem der Fehler schon behoben war. Das war der Fall beim großen Problem mit rpc.statd (`http://www.cert.org/advisories/CA-2000-17.html`) vor ein paar Jahren.

Es besteht eine weitgehende Zusammenarbeit zwischen den jeweiligen Sicherheitsteams der großen Linux-Distributionen. Bekannte Sicherheitsaktualisierungen werden selten (wenn nicht sogar nie) von den Anbietern der Distribution nicht eingespielt. Das Wissen um eine Sicherheitslücke wird niemals vor anderen Anbietern von Distributionen geheim gehalten, da die Ausbesserungen gewöhnlich vom Programmautor oder von CERT (`http://www.cert.org`) koordiniert werden. Das hat zur Folge, dass notwendige Sicherheitsaktualisierungen üblicherweise zur selben Zeit veröffentlicht werden. Damit ist die relative Sicherheit der verschiedenen Distributionen ziemlich ähnlich.

Einer großen Vorteile von Debian in Hinblick auf die Sicherheit ist die Leichtigkeit von Systemaktualisierungen mit `apt`. Hier sind ein paar andere Aspekte über die Sicherheit in Debian, die Sie berücksichtigen sollten:

- Debian bietet mehr Sicherheitswerkzeuge an als andere Distributionen. Vergleichen Sie dazu 'Sicherheitswerkzeuge in Debian' auf Seite 163.
- Debians Standardinstallation ist kleiner (weniger Funktionen) und daher sicherer. Andere Distributionen tendieren im Namen der Benutzerfreundlichkeit dazu, standardmäßig viele Dienst zu installieren, und manchmal sind diese nicht ordentlich konfiguriert (erinnern Sie sich an die Würmer Ramen und Lion (`http://www.sans.org/y2k/lion.htm`)). Debians Installation ist nicht so streng wie OpenBSD (Daemonen laufen standardmäßig nicht), aber es ist ein guter Kompromiss. [2]
- Debian stellt die besten Verfahren zur Sicherheit in Dokumenten wie diesem vor.

12.1.2 In Bugtraq gibt es viele Debian-Fehler. Heißt das, dass es sehr gefährdet ist?

Die Debian-Distribution enthält eine große und wachsende Zahl von Softwarepaketen, wahrscheinlich sogar mehr als mit vielen proprietären Betriebssystem geliefert wird. Je mehr Pakete installiert sind, desto größer ist die Möglichkeit von Sicherheitslücken in einem System.

[2] Ohne die Tatsache in Abrede zu stellen, dass einige Distributionen wie Red Hat oder Mandrake auch die Sicherheit bei ihrer Standardinstallation berücksichtigen, indem der Nutzer *Sicherheitsprofile* auswählen kann, oder Wizards verwendet werden, um beim Einrichten einer *Personal Firewall* zu helfen.

Immer mehr Menschen untersuchen den Quellcode, um Fehler zu entdecken. Es gibt viele Anweisungen im Zusammenhang mit Audits des Quellcodes von großen Softwarekomponenten, die in Debian enthalten sind. Immer wenn ein solcher Audit Sicherheitslücken aufdeckt, werden sie ausgebessert und eine Anweisung wird an Listen wie Bugtraq geschickt.

Fehler, die in der Debian-Distribution vorhanden sind, betreffen normalerweise auch andere Anbieter und Distributionen. Prüfen Sie einfach den "Debian specific: yes/no"-Abschnitt am Anfang jeder Anweisung (DSA).

12.1.3 Hat Debian irgendein Zertifikat für Sicherheit?

Die kurze Antwort: Nein.

Die lange Antwort: Zertifikate kosten Geld (besonders ein *seriöses* Sicherheitszertifikat). Niemand hat die Ressourcen aufgebracht, um Debian GNU/Linux beispielsweise mit irgendeinem Level des Common Criteria (`http://niap.nist.gov/cc-scheme/st/`) zertifizieren zu lassen. Wenn Sie daran interessiert sind, eine GNU/Linux-Distribution mit Sicherheitszertifikaten zu haben, stellen Sie uns die Ressourcen zur Verfügung, um dies möglich zu machen.

Es gibt im Moment mindestens zwei Linux-Distributionen, die mit verschiedenen EAL (`http://en.wikipedia.org/wiki/Evaluation_Assurance_Level`) Levels zertifiziert sind. Beachten Sie, dass einige CC-Tests im Linux Testing Project (`http://ltp.sourceforge.net`) vorhanden sind, welche in Debian durch `ltp` angeboten wird.

12.1.4 Gibt es irgendein Abhärtungsprogramm für Debian?

Ja. Bastille Linux (`http://www.bastille-linux.org`), das sich ursprünglich an anderen Linux-Distributionen (Red Hat und Mandrake) orientierte, funktioniert derzeit auch mit Debian. Es sind Maßnahmen eingeleitet, um Änderungen am Originalprogramm auch in das Debianpaket `bastille` einfließen zu lassen.

Manche Leute glauben jedoch, dass ein Absicherungsprogramm nicht die Notwendigkeit einer guten Administration ersetzt.

12.1.5 Ich möchte einen XYZ-Dienst laufen lassen. Welchen sollte ich benutzen?

Einer der größten Stärken von Debian ist die große Vielfalt von Paketen, die die gleichen Funktionen erfüllen (DNS-Server, Mail-Server, FTP-Server, Web-Server etc.). Das kann einen unerfahrenen Administrator verwirren, wenn er herausfinden will, welches Paket das richtige für ihn ist. Die beste Wahl hängt in der Balance zwischen Ihrem Bedürfnis nach Funktionalität und dem nach Sicherheit in der jeweiligen Situation ab. Im folgenden einige Fragen, die Sie sich stellen sollten, wenn Sie zwischen ähnlichen Paketen entscheiden müssen:

- Wird es noch vom Originalautor betreut? Wann war die letzte Veröffentlichung?

- Ist das Paket ausgereift? Die Versionsnummer sagt *nichts* darüber aus, wie ausgereift es ist. Versuchen Sie seine Geschichte nachzuvollziehen.
- Ist es von Fehlern durchsetzt? Gab es Sicherheits-Ankündigungen im Zusammenhang mit ihm?
- Stellt die Software die ganze Funktionalität zur Verfügung, die Sie benötigen? Bietet es mehr, als Sie wirklich brauchen?

12.1.6 Wie mache ich den Dienst XYZ unter Debian sicherer?

Sie werden in diesem Dokument Informationen über das Absichern von einigen Diensten (FTP, Bind) unter Debian GNU/Linux finden. Für Dienste die hier nicht abgedeckt werden, prüfen Sie die Programm-Dokumentation oder allgemeine Linux-Informationen. Die meisten Sicherheitshinweise für Unix-Systeme sind auch auf Debian anwendbar. So wird Dienst X unter Debian in den meisten Fällen wie in einer anderen Linux-Distribution (oder Un*x, was das betrifft) abgesichert.

12.1.7 Wie kann ich die Banner der Dienste entfernen?

Wenn Sie z.B. nicht wollen, dass Nutzer sich mit Ihrem POP3-Daemon verbinden und dadurch Informationen über Ihr System erlangen, sollten Sie das Banner, das der Dienst den Nutzern zeigt, entfernen (oder verändern). [3] Wie Sie das anstellen können. hängt von der Software ab, mit der Sie einen bestimmten Dienst betreiben. Für `postfix` stellen Sie beispielsweise das SMTP-Banner in `/etc/postfix/main.cf` ein:

```
smtpd_banner = $myhostname ESMTP $mail_name (Debian/GNU)
```

Andere Software kann nicht so leicht verändert werden. `ssh` muss neu kompiliert werden, um die angezeigte Version zu ändern. Stellen Sie sicher, dass sie nicht den ersten Teil des Banners (`SSH-2.0`) entfernen, da Clients ihn verwenden, um die von Ihrem Paket unterstützten Protokolle zu identifizieren.

12.1.8 Sind alle Debian Pakete sicher?

Das Sicherheitsteam von Debian kann nicht alle Pakete aus Debian auf potenzielle Sicherheitslücken hin analysieren, da es einfach nicht genug Ressourcen gibt, um für das gesamte Projekt ein Quellcodeaudit durchzuführen. Allerdings profitiert Debian von den Quellcode-Prüfungen durch die Originalautoren.

[3] Beachten Sie, dass das 'security by obscurity' ist und daher auf lange Sicht gesehen wahrscheinlich nicht der Mühe wert ist.

Tatsächlich könnte ein Debian-Entwickler in einem Paket einen Trojaner verbreiten, und es gibt keine Möglichkeit das nachzuprüfen. Sogar wenn es in einen Zweig von Debian eingeführt werden würde, wäre es unmöglich, alle möglichen Situationen abzudecken, in denen der Trojaner ausgeführt werden würde. Das ist der Grund, warum Debian eine *"Keine Gewährleistung"*-Klausel in seiner Lizenz hat.

Allerdings können Debian-Benutzer insofern Vertrauen fassen, dass der stabile Quellcode eine breite Prüfung hinter sich hat. Die meisten Probleme würden dabei durch Benutzung entdeckt. Es ist zu empfehlen, ungetestete Software auf kritischen Systemen zu installieren, wenn Sie nicht die notwendige Code-Prüfung vornehmen können. In jedem Fall gewährleistet der Aufnahmeprozess in die Distribution (mit digitalen Signaturen), dass im Falle von in die Distribution eingeschleusten Sicherheitsproblemen das Problem letztendlich zum Entwickler zurückgeführt werden kann. Das Debian-Projekt hat diese Angelegenheiten nie auf die leichte Schulter genommen.

12.1.9 Warum sind einige Log- und Konfigurationsdateien für die Welt lesbar? Ist das nicht unsicher?

Natürlich können Sie die Standardrecht von Debian auf Ihrem System abändern. Der aktuelle Grundsatz in Bezug auf Log- und Konfigurationsdateien besagt, dass sie für die Welt lesbar sind, *es sei denn*, sie enthalten sensible Informationen.

Seien Sie vorsichtig, wenn Sie Änderungen vornehmen:

- Prozesse könnten nicht mehr in der Lage sein, in Log-Dateien zu schreiben, wenn Sie ihre Rechte einschränken.
- Einige Anwendungen könnten nicht mehr funktionieren, wenn sie ihre Konfigurationsdatei nicht mehr lesen können. Wenn Sie zum Beispiel das Recht, für die Welt lesbar zu sein, von `/etc/samba/smb.conf` entfernen, kann das Programm `smbclient` nicht funktionieren, wenn es von einem normalen Nutzer ausgeführt wird.

FIXME: Check if this is written in the Policy. Some packages (i.e. ftp daemons) seem to enforce different permissions.

12.1.10 Warum hat /root/ (oder NutzerX) die Rechte 755?

Tatsächlich kann die gleiche Frage auch für jeden anderen Nutzer gestellt werden. Da Debians Standardinstallation *keine* Dateien unter diesem Verzeichnis abgelegt, sind keine sensiblen Informationen vorhanden, die geschützt werden müssten. Wenn Sie denken, dass diese Rechte für Ihr System zu locker sind, können Sie sie auf 750 einschränken. Für Nutzer sollten Sie 'Begrenzung des Zugangs zu Informationen anderer Nutzer' auf Seite 70 lesen.

Dieser Thread (`http://lists.debian.org/debian-devel/2000/debian-devel-200011/msg00783.html`) der Sicherheitsmailingliste von Debian hat weitere Ausführungen zu diesem Thema.

12.1.11 Nach der Installation von grsec oder einer Firewall bekomme ich viele Nachrichten auf der Konsole. Wie entferne ich sie?

Wenn Sie Nachrichten auf der Konsole empfangen und `/etc/syslog.conf` so eingerichtet haben, dass sie in Dateien oder auf ein spezielles TTY umgeleitet werden, sehen Sie vielleicht Nachrichten, die direkt an die Konsole geschickt werden.

Der Standardloglevel der Konsole ist bei jeden Kernel sieben, was bedeutet, dass alle Nachrichten mit einer niedrigeren Priorität auf der Konsole erscheinen werden. Für gewöhnlich haben Firewalls (die LOG-Regel) und einige andere Sicherheitswerkzeuge eine niedrigere Log-Priorität. Daher werden ihre Logs direkt an die Konsole geschickt.

Um die Nachrichten, die an die Konsole geschickt werden, nicht verringern, können Sie `dmesg` (Option `-n`, vergleichen Sie `dmesg(8)`) verwenden, das den Ringspeicher des Kernel untersucht und *steuert*. Damit das nach dem nächsten Neustart in Ordnung ist, ändern Sie in `/etc/init.d/klogd`

```
KLOGD=""
```

zu

```
KLOGD="-c 4"
```

ab.

Verwenden Sie eine niedrigere Nummer für `-c`, wenn Sie immer noch unerwünschte Nachrichten sehen. Eine Beschreibung der verschiedenen Loglevels befindet sich in `/usr/include/sys/syslog.h`:

```
#define LOG_EMERG       0       /* system is unusable */
#define LOG_ALERT       1       /* action must be taken immediately */
#define LOG_CRIT        2       /* critical conditions */
#define LOG_ERR         3       /* error conditions */
#define LOG_WARNING     4       /* warning conditions */
#define LOG_NOTICE      5       /* normal but significant condition */
#define LOG_INFO        6       /* informational */
#define LOG_DEBUG       7       /* debug-level messages */
```

12.1.12 Benutzer und Gruppen des Betriebssystems

Sind alle Systemnutzer notwendig?

Ja und nein. Debian wird mit einigen vordefinierten Nutzern (User-ID (UID) < 99, beschrieben in der Debian Policy (`http://www.de.debian.org/doc/debian-policy/`) oder in `/usr/share/doc/base-passwd/README`) geliefert. Dadurch wird die Installation einiger

Dienste erleichtert, für die es notwendig ist, unter einem passenden Nutzer/UID zu laufen. Wenn Sie nicht vorhaben, neue Dienste zu installieren, können Sie die Nutzer entfernen, denen keine Dateien auf Ihrem System gehören und die keine Dienste laufen lassen. Unabhängig davon ist das Standardverhalten in Debian, dass UIDs von 0 bis 99 reserviert sind und UIDs von 100 bis 999 von Paketen bei der Installation erstellt werden und gelöscht werden, wenn das Pakete vollständig gelöscht wird (purge) wird.

Nutzer, denen keine Dateien gehören, finden Sie leicht mit dem folgenden Kommando (führen Sie es als Root aus, da ein normaler Nutzer nicht genügend Zugriffsrechte haben könnte, um einige sensible Verzeichnisse zu durchsuchen):

```
cut -f 1 -d : /etc/passwd | \
while read i; do find / -user "$i" | grep -q . && echo "$i"; done
```

Diese Nutzer kommen aus dem Paket `base-passwd`. Sie finden Informationen über die Behandlung dieser Nutzer unter Debian in der Dokumentation des Pakets. Es folgt nun eine Liste der Standardnutzer (mit einer entsprechenden Gruppe):

- root: Root ist (typischerweise) der Superuser.
- daemon: Einige unprivilegierte Daemonen, die Dateien auf die Festplatte schreiben müssen, laufen als daemon.daemon (z.B. `portmap`, `atd`, wahrscheinlich noch andere). Daemonen, die keine eigenen Dateien besitzen müssen, können stattdessen als nobody.nogroup laufen. Komplexere oder sicherheitsbewusste Daemonen laufen als eigenständige Nutzer. Der Nutzer daemon ist auch praktisch für lokal installierte Daemons.
- bin: aus historischen Gründen beibehalten.
- sys: das gleiche wie bei bin. Jedoch gehören /dev/vcs* und `/var/spool/cups` der Gruppe sys.
- sync: Die Shell des Nutzers sync ist `/bin/sync`. Wenn das Passwort auf etwas leicht zu ratendes gesetzt wurde (zum Beispiel “”), kann jeder das System von der Konsole aus synchronisieren lassen, auch wenn er kein Konto hat.
- games: Viele Spiele sind SETGID games, so dass sie ihre Highscore-Dateien schreiben können. Dies wird in der Policy erklärt.
- man: Das Programm man läuft (manchmal) als Nutzer man, damit es Cat-Seiten nach `/var/cache/man` schreiben kann.
- lp: Wird von Druck-Daemonen benutzt.
- mail: Mailboxen unter `/var/mail` gehören der Gruppe mail, wie in der Policy erklärt wird. Der Nutzer und die Gruppe werden auch von verschiedene MTAs zu anderen Zwecken benutzt.

- news: Verschiedene News-Server und ähnliche Programme (zum Beispiel `suck`) benutzen den Nutzer und die Gruppe news auf unterschiedliche Weise. Dateien im news-Spool gehören häufig dem Nutzer und der Gruppe news. Programme wie `inews`, die man benutzen kann, um News zu posten, sind normalerweise SETGID news.
- uucp: Der Nutzer uucp und die Gruppe uucp werden vom UUCP-Subsystem benutzt. Ihnen gehören Spool- und Konfigurationsdateien. Nutzer in der Gruppe uucp können uucico aufrufen.
- proxy: Wie Daemon wird dieser Nutzer und diese Gruppe von manchen Daemonen (insbesondere Proxy-Daemonen) verwendet, die keine spezielle User-ID haben, aber eigene Dateien besitzen müssen. Zum Beispiel wird die Gruppe proxy von `pdnsd` benutzt, und `squid` läuft als Nutzer proxy.
- majordom: `Majordomo` hat auf Debian-Systemen aus historischen Gründen eine statisch zugewiesene UID. Auf neuen Systemen wird sie nicht installiert.
- postgres: `Postgresql`-Datenbanken gehören diesem Nutzer und dieser Gruppe. Alle Dateien in `/var/lib/postgresql` gehören diesem Nutzer, um anständige Sicherheit zu gewährleisten.
- www-data: Einige Web-Server laufen als www-data. Web-Inhalte sollten *nicht* diesem Nutzer gehören, andernfalls wäre ein kompromittierter Web-Server in der Lage, eine Web-Seite zu überschreiben. Daten, die der Web-Server schreibt, einschließlich Log-Dateien, gehören www-data.
- backup: So können Backup-/Wiederherstellungszuständigkeiten lokal an irgendjemanden ohne volle Root-Zugriff delegiert werden.
- operator: operator ist historisch (und praktisch) das einzige 'Nutzer'-Konto, in das man sich entfernt einloggen kann, und das nicht von NIS/NFS abhängt.
- list: Mailinglisten-Archive und Daten gehören diesem Nutzer und dieser Gruppe. Manche Mailinglisten-Programme laufen auch unter diesem Nutzer.
- irc: Wird von irc-Daemonen benutzt. Ein statisch zugewiesener Nutzer wird nur wegen eines Fehlers in `ircd` benötigt, das beim Start SETUID() auf sich selbst für eine bestimmte UID ausführt.
- gnats.
- nobody, nogroup: Daemonen die keine eigenen Dateien haben laufen als Nutzer nobody und Gruppe nogroup. Demzufolge sollten keine Dateien auf dem gesamten System diesem Nutzer oder dieser Gruppe gehören.

Andere Gruppe, die keinen dazugehörigen Benutzer haben:

- adm: Die Gruppe adm wird zu Zwecken der Überwachung benutzt. Mitglieder dieser Gruppe können viele Dateien in `/var/log` lesen und die xconsole benutzen. `/var/log` war früher einmal `/usr/adm` (und später `/var/adm`), daher der Name dieser Gruppe.

- tty: TTY-Geräte gehören dieser Gruppe. Die Befehle write und wall benutzen dies, um auf die TTYs anderer Leute zu schreiben.
- disk: Roh-Zugriff auf Festplatten. Größtenteils äquivalent zum Root-Zugriff.
- kmem: /dev/kmem und ähnliche Dateien sind von dieser Gruppe lesbar. Dies ist größtenteils ein Relikt aus BSD. Aber jedes Programm, dass Lese-Zugriff auf den Systemspeicher braucht, kann so SETGID kmem gemacht werden.
- dialout: Voller und direkter Zugriff auf serielle Schnittstellen. Mitglieder dieser Gruppen können Modems rekonfigurieren, sich irgendwo einwählen, usw.
- dip: Der Name der Gruppe steht für "Dial-up IP". Mitglied der Gruppe dip zu sein erlaubt Ihnen Programme wie `ppp`, `dip`, `wvdial` usw. zu benutzen, um eine Verbindung herzustellen. Die Nutzer in dieser Gruppe können das Modem nicht konfigurieren. Sie können lediglich Programme aufrufen, die es benutzen.
- fax: Erlaubt es den Mitgliedern, Fax-Software zu benutzen, um Faxe zu senden und zu empfangen.
- voice: Voicemail, nützlich für Systeme, die Modems als Anrufbeantworter benutzen.
- cdrom: Diese Gruppe kann dazu benutzt werden, einer bestimmen Menge von Nutzern Zugriff auf CD-ROM-Laufwerke zu geben.
- floppy: Diese Gruppe kann dazu benutzt werden, einer bestimmen Menge von Nutzern Zugriff auf Diskettenlaufwerke zu geben.
- tape: Diese Gruppe kann dazu benutzt werden, einer bestimmen Menge von Nutzern Zugriff auf Bandlaufwerke zu geben.
- sudo: Mitglieder dieser Gruppe müssen ihr Passwort nicht eingeben, wenn sie `sudo` benutzen. Siehe `/usr/share/doc/sudo/OPTIONS`.
- audio: Diese Gruppe kann dazu benutzt werden, einer bestimmen Menge von Nutzern Zugriff auf jedes Audiogerät zu geben.
- src: Dieser Gruppe gehören die Quellcodes, einschließlich der Dateien in `/usr/src`. Sie kann benutzt werden, um einem bestimmten Nutzern die Möglichkeit zu bieten, Quellcode des Systems zu verwalten.
- shadow: `/etc/shadow` ist von dieser Gruppe lesbar. Einige Programme, die auf diese Datei zugreifen müssen, sind SETGID shadow.
- utmp: Diese Gruppe kann nach `/var/run/utmp` und ähnlichen Dateien schreiben. Programme, die darin schreiben können müssen, sind SETGID utmp.
- video: Diese Gruppe kann dazu benutzt werden, einer bestimmen Menge von Nutzern Zugriff auf ein Videogerät zu geben.

- staff: Erlaubt Nutzern lokale Modifikationen am System vorzunehmen (`/usr/local`, `/home`), ohne dass sie Root-Privilegien bräuchten. Vergleichen Sie sie mit "adm", die sich mehr auf Überwachung/Sicherheit bezieht.

- users: Während Debian-Systeme standardmäßig das System einer privaten Nutzergruppe (jeder Nutzer hat seine eigene Gruppe) benutzen, ziehen es manche vor, ein traditionelleres Gruppen-System zu verwenden. In diesem System ist jeder Nutzer Mitglied dieser Gruppe.

Ich entfernte einen Systembenutzer! Wie kann ich dies rückgängig machen?

Wenn Sie einen Systembenutzer entfernt und kein Backup Ihrer `password`- und `group`-Dateien haben, können Sie versuchen, diesen mittels `update-passwd` (vergleichen Sie `update-passwd(8)`) wiederherzustellen.

Was ist der Unterschied zwischen den Gruppen adm und staff?

Die Gruppe 'adm' besteht üblicherweise aus Administratoren. Die Rechte dieser Gruppe erlauben es ihnen, Log-Dateien zu lesen, ohne `su` benutzen zu müssen. Die Gruppe 'staff' ist gewöhnlich für Kundendienst- und Junioradministratoren bestimmt und gibt ihnen die Möglichkeit, Dinge in `/usr/local` zu erledigen und Verzeichnisse in `/home` anzulegen.

12.1.13 Warum gibt es eine neue Gruppe, wenn ich einen neuen Nutzer anlege? (Oder warum gibt Debian jedem Nutzer eine eigene Gruppe?)

Das Standardverhalten von Debian ist, dass jeder Nutzer seine eigene, persönliche Gruppe hat. Das traditionelle UN*X-Modell weist alle Benutzer der Gruppe *users* zu. Zusätzliche Gruppe werden erstellt, um den Zugang zu gemeinsam genutzten Dateien, die mit verschiedenen Projektverzeichnissen verbunden sind, einzuschränken. Die Dateiverwaltung wurde schwierig, wenn ein einzelner Nutzer an verschiedenen Projekten arbeitete, da, wenn jemand eine Datei erstellte, diese mit der primären Gruppe des Erstellers (z.B. 'users') verbunden war.

Das Modell von Debian löst dieses Problem, indem es jedem Nutzer seine eigene Gruppe zuweist. So wird mit einer korrekten Umask (0002) und mit dem SETGID-Bit für ein Projektverzeichnis den Dateien, die in diesem Verzeichnis erstellt werden, automatisch die richtige Gruppe zugewiesen. Das erleichtert die Arbeit von Menschen, die an verschiedenen Projekten arbeiten, da sie nicht die Gruppe oder Umasks ändern müssen, wenn sie mit gemeinsam genutzten Dateien arbeiten.

Sie können allerdings dieses Verhalten verändern, indem Sie `/etc/adduser.conf` modifizieren. Ändern Sie die Variable *USERGROUPS* auf 'no' ab. Dadurch wird keine neue Gruppe erstellt, wenn ein neuer Nutzer angelegt wird. Sie sollten auch *USERS_GID* die GID der Gruppe zuweisen, der alle Nutzer angehören.

12.1.14 Fragen über Dienste und offene Ports

Warum werden Dienste während der Installation aktiviert?

Das ist der Annäherung an das Problem, auf der einen Seite sicherheitsbewusst und auf der anderen Seite benutzerfreundlich zu sein. Anders als OpenBSD, das alle Dienste abschaltet, bis sie vom Administrator aktiviert werden, aktiviert Debian GNU/Linux alle installierten Dienste, bis sie abgeschaltet werden (siehe dazu 'Daemons abschalten' auf Seite 40). Immerhin haben Sie den Dienst installiert, oder?

Es gab viele Diskussionen auf Debian-Mailinglisten (sowohl auf debian-devel als auch auf debian-security) darüber, welches die bessere Vorgehensweise für eine Standardinstallation ist. Jedoch gab es bisher (10. März 2002) keinen Konsens.

Kann ich `inetd` entfernen?

`Inetd` ist nicht leicht zu entfernen, da `netbase` von dem Paket abhängt, das es enthält (`netkit-inetd`). Wenn Sie es entfernen wollen, können Sie es entweder abschalten (siehe 'Daemons abschalten' auf Seite 40) oder das Paket entfernen, indem Sie das Paket `equivs` benutzen.

Warum muss ich Port 111 offen haben?

Port 111 ist sunrpcs Portmapper und wird standardmäßig bei der Grundinstallationen eines Debian-Systems eingerichtet, da es keine Möglichkeit gibt herauszubekommen, wann ein Programm eines Nutzers RPC gebrauchen könnte, um korrekt zu arbeiten. Jedenfalls wird es meistens von NFS benutzt. Wenn Sie kein NFS benutzen, entfernen Sie es, wie in 'Sichern von RPC-Diensten' auf Seite 120 erklärt.

In Versionen des Pakets `portmap` später als 5-5 können Sie sogar den Portmapper installieren, aber ihn nur auf dem Localhost lauschen lassen (dazu müssen Sie `/etc/default/portmap` verändern).

Wozu ist der `identd` (Port 113) da?

Der Dienst Identd ist ein Authentisierungdienst, der den Besitzer einer bestimmten TCP/IP-Verbindung zu einem entfernten Server, der die Verbindung annimmt, identifiziert. Wenn ein Benutzer sich mit einem entfernten Host verbindet, schickt `inetd` auf dem entfernten Host üblicherweise eine Anfrage an Port 113 zurück, um Informationen über den Besitzer herauszufinden. Er wird häufig von Mail-, FTP- und IRC-Servern eingesetzt. Er kann auch dazu verwendet werden, um einen Nutzer Ihres lokalen Systems, der ein entferntes System angreift, aufzuspüren.

Es gab ausführliche Diskussionen über die Sicherheit von `identd` (siehe in den Archiven der Mailingliste (`http://lists.debian.org/debian-security/2001/`

`debian-security-200108/msg00297.html`). Im Allgemeinen ist `identd` auf Multi-User-Systemen nützlicher als auch einer Workstation mit nur einem Benutzer. Wenn Sie keine Verwendung von ihn haben, sollten Sie ihn abschalten, damit Sie keinen Dienst für die Außenwelt offen lassen. Wenn Sie sich entscheiden, den identd-Port mit einer Firewall zu blockieren, benutzen Sie *bitte* die Regel 'reject' und nicht die Regel 'deny', da andernfalls eine Verbindung zu einem Server, die `identd` verwendet, bis zu einer Zeitüberschreitung hängen bleiben wird (lesen Sie dazu reject or deny issues (`http://logi.cc/linux/reject_or_deny.php3`)).

Ich habe Dienste, die die Ports 1 und 6 verwenden. Welche sind das und wie kann ich sie entfernen?

Sie führen den Befehl `netstat -an` aus und erhalten Folgendes:

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
raw        0      0 0.0.0.0:1               0.0.0.0:*               7
-
raw        0      0 0.0.0.0:6               0.0.0.0:*               7
-
```

Sie sehen *nicht* Prozesse, die auf dem TCP/UDP-Port 1 und 6 lauschen. Tatsächlich sehen Sie einen Prozess, der auf einem *Raw*-Socket für Protokoll 1 (ICMP) und 6 (TCP) lauscht. Ein solches Verhalten ist für Trojaner und einige Systeme zur Eindringlingserkennung wie `iplogger` und `portsentry` üblich. Wenn Sie diese Pakete besitzen, löschen Sie sie einfach. Falls nicht, versuchen Sie mit netcats Option `-p` (Prozess) herauszufinden, welcher Prozess diese Lauscher betreibt.

Ich habe festgestellt, dass ich den folgenden Port (XYZ) offen habe. Kann ich ihn schließen?

Ja, natürlich. Die Ports, die Sie offen lassen, hängen von Ihrem individuellen Regelwerk bezüglich öffentlich zugänglicher Dienste ab. Prüfen Sie, ob sie von `inetd` (siehe 'Abschalten von `inetd` oder seinen Diensten' auf Seite 41) oder von anderen installierten Paketen geöffnet werden, und leiten Sie passende Maßnahmen ein (d.h. konfigurieren Sie inetd, entfernen Sie das Paket, verhindern Sie, dass der Dienst beim Booten gestartet wird).

Hilft das Löschen von Diensten aus `/etc/services`, um meinen Rechner abzusichern.

Nein, `/etc/services` stellt nur eine Verbindung zwischen virtuellem Namen und Portnummer her. Das Entfernen von Namen aus dieser Datei verhindert (üblicherweise) nicht, dass ein Dienst gestartet wird. Manche Daemonen starten vielleicht nicht, wenn `/etc/services` verändert wurde, aber das ist nicht die Norm. Um einen Dienst richtig abzuschalten, sehen Sie sich 'Daemons abschalten' auf Seite 40 an.

12.1.15 Allgemeine Sicherheitsfragen

Ich habe mein Passwort vergessen und kann auf das System nicht mehr zugreifen!

Die nötigen Schritte, um wieder Zugriff erhalten, hängen davon ab, ob Sie die vorgeschlagene Prozedur zum Absichern von `lilo` und BIOS durchgeführt haben oder nicht.

Wenn Sie beides eingeschränkt haben, müssen Sie im BIOS erlauben, von anderen Medien als der Festplatte zu booten, bevor Sie weitermachen können. Wenn Sie auch Ihr BIOS-Passwort vergessen haben, müssen Sie Ihr BIOS zurücksetzen. Dazu öffnen Sie das PC-Gehäuse und entfernen die BIOS-Batterie.

Sobald Sie das Booten von CD-ROM oder Diskette eingeschaltet haben, sollten Sie Folgendes ausprobieren:

- Booten Sie von eine Rettungsdiskette und starten den Kernel.
- Wechseln Sie mit Alt+F2 auf eine virtuelle Konsole.
- Mounten Sie die Partition, auf der sich Ihr /root befindet.
- Editieren Sie (auf der Rettungsdiskette von Debian 2.2 befindet sich `ae`, Debian 3.0 enthält `nano-tiny`, der `vi` ähnelt) die Datei `/etc/shadow` und ändern Sie die Zeile:

  ```
  root:asdfjl290341274075:XXXX:X:XXXX:X::: (X=irgendeine Ziffer)
  ```

  in:

  ```
  root::XXXX:X:XXXX:X:::
  ```

Dies entfernt das vergessene Root-Passwort, das sich im ersten durch Doppelpunkte abgetrennten Feld nach dem Nutzernamen befand. Speichern Sie die Datei ab, starten Sie das System neu und melden Sie sich als Root mit einem leeren Passwort an. Dies wird funktionieren, außer wenn Sie Ihr System etwas sicherer eingestellt haben, d.h. wenn Sie nicht erlauben, dass Nutzer leere Passwörter haben, oder dass Root sich auf einer Konsole einloggen kann.

Falls Sie derartige Maßnahmen getroffen haben, müssen Sie im Single-User-Modus starten. Wenn Sie LILO eingeschränkt haben, müssen `lilo` erneut ausführen, nachdem Sie das Root-Passwort zurückgesetzt haben. Das ist ziemlich verzwickt, da Ihre `/etc/lilo.conf` verändert werden muss, da das Root-Dateisystem (/) eine RAM-Disk und keine echte Festplatte ist.

Sobald LILO nicht mehr eingeschränkt ist, versuchen Sie Folgendes:

- Drücken Sie Alt, Shift oder Steuerung (Control), kurz bevor das BIOS seine Arbeit beendet hat, und Sie sollten nun einen LILO-Prompt erhalten.
- Geben Sie am Prompt `linux single`, `linux init=/bin/sh` oder `linux 1` ein.

- Sie erhalten einen Shell-Prompt im Single-User-Modus (Sie werden nach dem Passwort gefragt, aber das kennen Sie jetzt ja)
- Binden Sie die Root-Partition (/) im Schreib/Lese-Modus neu ein, indem Sie den Befehl mount verwenden:

```
mount -o remount,rw /
```

- Ändern Sie das Superuser-Passwort mit `passwd` (da Sie der Superuser sind, werden Sie nicht nach dem alten Passwort gefragt).

12.1.16 Wie muss ich vorgehen, wenn ich meinen Nutzern einen Dienst anbieten möchte, ihnen aber kein Shell-Konto geben will?

Wenn Sie zum Beispiel einen POP-Dienst anbieten wollen, müssen Sie nicht für jeden zugreifenden Benutzer ein Konto anlegen. Am besten setzen Sie hierzu eine Authentifizierung, die auf Verzeichnisses basiert, durch einen externen Dienst (wie Radius, LDAP oder eine SQL-Datenbank) ein. Installieren Sie einfach die gewünschte PAM-Bibliothek (`libpam-radius-auth`, `libpam-ldap`, `libpam-pgsql` oder `libpam-mysql`), lesen Sie die Dokumentation (Einsteiger sehen bitte unter 'Nutzerauthentifizierung: PAM' auf Seite 58 nach) und konfigurieren Sie den PAM-nutzenden Dienst, so dass er Ihren Backend benutzt. Bearbeiten Sie dazu die dem Dienst entsprechenden Dateien unter `/etc/pam.d/` und ändern die folgenden Zeile von

```
auth    required        pam_unix_auth.so shadow nullok use_first_pass
```

beispielsweise für ldap zu:

```
auth    required        pam_ldap.so
```

Im Fall von LDAP-Verzeichnissen liefern manche Dienste LDAP-Schemata mit, die Sie Ihrem Verzeichnis hinzufügen können, um eine LDAP-Authentifizierung zu benutzen. Wenn Sie relationale Datenbanken benutzen, gibt es einen nützlichen Trick: Benutzen Sie die Klausel *where*, wenn Sie die PAM-Module konfigurieren. Wenn Sie beispielsweise eine Datenbank mit der folgenden Tabelle haben:

```
(user_id,user_name,realname,shell,password,uid,gid,homedir,sys,pop,imap,ftp
```

Wenn Sie die Attribute der Dienste zu Boolean-Feldern machen, können Sie sie verwenden, um den Zugang zu den verschiedenen Diensten zu erlauben oder zu verbieten. Sie müssen dazu nur die geeigneten Zeilen in folgende Dateien einfügen:

- `/etc/pam.d/imap`:`where=imap=1`.
- `/etc/pam.d/qpopper`:`where=pop=1`.
- `/etc/nss-mysql*.conf`:`users.where_clause = user.sys = 1;`.
- `/etc/proftpd.conf`:`SQLWhereClause "ftp=1"`.

12.2 Mein System ist angreifbar! (Sind Sie sich sicher?)

12.2.1 Der Scanner X zur Einschätzung der Verwundbarkeit sagt, dass mein Debian-System verwundbar wäre?

Viele Scanner zur Einschätzung der Verwundbarkeit liefern falsche Positivmeldungen, wenn sie auf Debian-Systemen verwendet werden. Das liegt daran, dass sie nur die Version eines Softwarepakets überprüfen, um herauszufinden, ob es verwundbar ist. Sie prüfen nicht, ob tatsächlich eine Sicherheitslücke vorhanden ist. Da Debian nicht die Version einer Software ändert, wenn ein Paket repariert wird (häufig werden Ausbesserungen an neueren Veröffentlichungen zurückportiert), neigen einige Werkzeuge dazu zu denken, dass ein aktualisiertes Debian-System verwundbar ist, auch wenn das nicht der Fall ist.

Wenn Sie denken, dass Ihr System auf dem aktuellen Stand der Sicherheitsaktualisierungen ist, sollten Sie die Querverweise zu den Datenbanken mit Sicherheitslücken, in denen die DSAs veröffentlicht sind (vergleichen Sie dazu 'Debian-Sicherheits-Ankündigungen' auf Seite 138), verwenden, um falsche Positive auszusondern, wenn das Programm, das Sie verwenden, CVE-Referenzen enthält.

12.2.2 Ich habe in meinen Logfiles einen Angriff gesehen: Ist mein System kompromittiert?

Ein Hinweis auf einen Angriff heißt nicht notwendigerweise, dass Ihr System gehackt wurde. Leiten Sie die üblichen Schritte ein, um festzustellen, ob das System kompromittiert wurde (siehe 'Nach einer Kompromittierung (Reaktion auf einem Vorfall)' auf Seite 195). Beachten Sie auch, dass Spuren des Angriffs in den Logs mitunter bedeuten, dass Ihr System hierfür bereits angreifbar sind (ein entschlossener Angreifer könnte übrigens auch andere Angriffe, als die von Ihnen beobachteten durchgeführt haben).

12.2.3 Ich habe in meinen Logs merkwürdige "MARK"-Einträge gefunden. Wurde ich gehackt?

Sie können die folgenden Zeilen in Ihren System-Logs finden:

```
Dec 30 07:33:36 debian -- MARK --
Dec 30 07:53:36 debian -- MARK --
Dec 30 08:13:36 debian -- MARK --
```

Dies stellt keinen Hinweis auf eine Kompromittierung dar, obwohl Nutzer, die von einer Debian-Release wechseln, es vielleicht merkwürdig finden. Wenn Ihr System keine große Last (oder nicht viele aktive Dienste) hat, können diese Zeilen in alle Logs auftauchen. Dies ist ein Hinweis, dass Ihr `syslogd`-Daemon richtig läuft. Aus `syslogd(8)`:

```
-m interval
       Der Syslogd protokolliert regelmäßig einen
       Zeitstempel. Der voreingestellte Abstand zwischen zwei --
       MARK -- Zeilen ist 20 Minuten. Er kann mit dieser Option
       geändert werden. Setzen Sie den Abstand auf Null, um
       die Zeitstempel komplett abzuschalten.
```

12.2.4 Ich habe Nutzer gefunden, die laut meinen Logfiles 'su' benutzen: Bin ich kompromittiert?

Sie könnten in Ihren Logdateien Zeilen wie die folgenden finden:

```
Apr  1 09:25:01 server su[30315]: + ??? root-nobody
Apr  1 09:25:01 server PAM_unix[30315]: (su) session opened for user nobody
```

Seien Sie nicht zu besorgt. Prüfen Sie, ob dies durch einen `Cron`-Job hervorgerufen wird (normalerweise `/etc/cron.daily/find` oder `logrotate`):

```
$ grep 25 /etc/crontab
25 6    * * *   root    test -e /usr/sbin/anacron || run-parts --report
/etc/cron.daily
$ grep nobody /etc/cron.daily/*
find:cd / && updatedb --localuser=nobody 2>/dev/null
```

12.2.5 Ich habe 'possible SYN flooding' in meinen Logs entdeckt: Werde ich angegriffen?

Sie sehen Einträge wie diese in Ihren Logs:

```
May 1 12:35:25 linux kernel: possible SYN flooding on port X. Sending cooki
May 1 12:36:25 linux kernel: possible SYN flooding on port X. Sending cooki
May 1 12:37:25 linux kernel: possible SYN flooding on port X. Sending cooki
May 1 13:43:11 linux kernel: possible SYN flooding on port X. Sending cooki
```

Überprüfen Sie mit `netstat`, ob es eine große Anzahl von Verbindungen zum Server gibt. Zum Beispiel:

```
linux:~# netstat -ant | grep SYN_RECV | wc -l
   9000
```

Dies ist ein Anzeichen, dass ein Denial-of-Service-Angriff (DoS) auf den Port X Ihres Systems (am wahrscheinlichsten gegen einen öffentlichen Dienst wie Ihr Web- oder Mailserver). Sie

sollten TCP-Syncookies in Ihrem Kernel einschalten, siehe 'Konfiguration von Syncookies' auf Seite 87. Beachten Sie, dass ein DoS-Angriff Ihr Netzwerk überfluten kann, auch wenn Sie verhindern können, dass er Ihr System zum Absturz bringt. [4] Der einzige effektive Weg, diesen Angriff abzuwehren, ist, mit Ihrem Netzprovider in Verbindung zu treten.

12.2.6 Ich habe seltsame Root-Sessions in meinen Logs entdeckt: Wurde ich gehackt?

Sie sehen folgende Art von Einträgen in der Datei `/var/log/auth.log`:

```
May 2 11:55:02 linux PAM_unix[1477]: (cron) session closed for user root
May 2 11:55:02 linux PAM_unix[1476]: (cron) session closed for user root
May 2 12:00:01 linux PAM_unix[1536]: (cron) session opened for user root by
(UID=0)
May 2 12:00:02 linux PAM_unix[1536]: (cron) session closed for user root
```

Sie kommen von einem ausgeführten `Cron`-Job (in unserem Beispiel alle fünf Minuten). Um herauszufinden, welches Programm für diese Jobs verantwortlich ist, überprüfen Sie die Einträge in `/etc/crontab`, `/etc/cron.d`, `/etc/crond.daily` und Roots `crontab` in `/var/spool/cron/crontabs`.

12.2.7 Ich bin Opfer eines Einbruchs, was soll ich jetzt tun?

Es gibt mehrere Schritte, die Sie bei einem Einbruch durchführen sollten:

- Prüfen Sie, ob Ihr System auf dem aktuellen Stand der Sicherheitsaktualisierungen für veröffentlichte Verwundbarkeiten ist. Wenn Ihr System verwundbar ist, hat die die Möglichkeit, dass Ihr System tatsächlich gehackt wurde, erhöht. Die Wahrscheinlichkeit steigt weiter an, wenn die Sicherheitslücke schon eine Zeit lang bekannt ist, da üblicherweise mehr Tätigkeit mit älteren Verwundbarkeiten besteht. Hier ist ein Link zu SANS Top 20 Security Vulnerabilities (`http://www.sans.org/top20/`).
- Lesen Sie dieses Dokument, besonders den Abschnitt 'Nach einer Kompromittierung (Reaktion auf einem Vorfall)' auf Seite 195.
- Fragen Sie nach Hilfe. Sie können die Mailingliste debian-security benutzen und um Rat fragen, wie Sie Ihr System wiederherstellen oder patchen.
- Benachrichtigen Sie Ihren lokalen CERT (`http://www.cert.org`) (wenn einer existiert, ansonsten sollten Sie sich vielleicht direkt mit CERT in Verbindung setzen). Das könnte Ihnen helfen (vielleicht aber auch nicht), aber wenigstens wird CERT über laufende Angriffe informiert. Diese Information ist sehr wertvoll, um herauszufinden, welche Werkzeuge und Angriffsarten von der *Blackhat*-Community verwendet werden.

[4] Da keine Datei-Deskriptoren mehr vorhanden sind, könnte das System nicht mehr antworten, bis das Zeitlimit der TCP-Verbindungen überschritten wurde.

12.2.8 Wie verfolge ich einen Angriff zurück?

Sie können einen Angriff zu seinem Ursprung zurückverfolgen, indem Sie die Logs (wenn sie nicht geändert wurden) mit Hilfe eines Systems zur Eindringlingserkennung (siehe 'Aufsetzen einer Eindringlingserkennung' auf Seite 187), `traceroute`, `whois` oder ähnlicher Werkzeuge (einschließlich forensischer Analyse) durchsehen. Wie Sie auf diese Informationen reagieren und was *Sie* als Angriff betrachten, hängt ausschließlich von Ihren Sicherheitsrichtlinien ab. Ist ein einfacher Scan ein Angriff? Ist die Prüfung auf eine Verwundbarkeit ein Angriff?

12.2.9 Das Programm X in Debian ist angreifbar – was soll ich tun?

Nehmen Sie sich zuerst einen Augenblick Zeit, um zu schauen, ob die Sicherheitslücke in öffentlichen Sicherheitsmailinglisten (wie Bugtraq) oder anderen Foren bekannt gemacht wurde. Das Sicherheitsteam von Debian ist hinsichtlich dieser Listen auf dem Laufenden, daher könnte ihm dieses Problem bereits bekannt sein. Leiten Sie keine weiteren Maßnahmen ein, wenn Sie schon eine Bekanntmachung auf `http://security.debian.org` sehen.

Wenn anscheinend keine Informationen veröffentlicht wurden, schicken Sie bitte eine E-Mail zu den betroffenen Paketen mit einer detaillierten Beschreibung der Verwundbarkeit (Code, der dies bestätigt, ist auch in Ordnung) an team@security.debian.org (`mailto:team@security.debian.org`). Dort erreichen Sie das Sicherheitsteam von Debian.

12.2.10 Laut der Versionsnummer eines Paketes läuft bei mir immer noch eine angreifbare Version!

Statt auf neue Veröffentlichung zu aktualisieren, portiert Debian sicherheitsrelevante Korrekturen zu der Version zurück, die in der Stable-Veröffentlichung enthalten ist. Der Grund dafür ist, dass sicher gegangen werden soll, dass die Stable-Veröffentlichung so wenig wie möglich verändert wird. Damit wird verhindert, dass sich Dinge als Folge einer Sicherheitskorrektur unerwartet ändern oder kaputt gehen. Ob Sie eine sichere Version eines Paketes benutzen, stellen Sie fest, indem Sie das Changelog des Paketes durchsehen, oder indem Sie die exakte Versionsnummer (ursprüngliche Version -slash- Debian-Release) mit der Nummer aus der Debian-Sicherheits-Ankündigung (DSA) vergleichen.

12.2.11 Spezielle Software

`Proftpd` ist für einen Denial-of-Service-Angriff anfällig.

Fügen Sie Ihrer Konfigurationsdatei `DenyFilter \*.*/` hinzu. Mehr Informationen entnehmen Sie `http://www.proftpd.org/critbugs.html`.

Nach der Installation von `portsentry` sind viele Ports offen.

Dies ist nur die Art und Weise, wie `portsentry` arbeitet. Es öffnet etwas zwanzig ungenutzte Ports und versucht so, Port-Scans zu entdecken.

12.3 Fragen zum Sicherheitsteam von Debian

Diese Informationen stammen aus dem Debian Sicherheits-FAQ (http://www.de.debian.org/security/faq). Es umfasst Informationen zum 19. November und beantwortet auch andere häufige Fragen aus der Mailingliste debian-security.

12.3.1 Was ist eine Debian-Sicherheits-Ankündigung (Debian Security Advisory, DSA)?

Das Debian-Sicherheitsteam (siehe unten) weist darin auf die Entdeckung und Korrektur von sicherheitsrelevanten Verwundbarkeiten in einem Paket von Debian GNU/Linux hin. Digital unterschriebene DSAs werden an öffentliche Mailinglisten gesendet und auf der Webseite von Debian veröffentlicht (sowohl auf der Hauptseite als auch unter Sicherheits-Informationen (http://www.de.debian.org/security/)).

DSAs enthalten Informationen über die beeinträchtigten Pakete, den entdeckten Sicherheitsmangel und wo man die aktualisierten Pakete bekommt (und ihre MD5-Summen).

12.3.2 Die digitale Signatur eines Debian-Anweisung wird nicht korrekt verifiziert!

Dies ist wahrscheinlich ein Problem auf Ihrer Seite. Die Liste debian-security-announce (http://www.de.debian.org/security/faq) besitzt einen Filter, der es nur erlaubt, Nachrichten mit einer korrekten Signatur eines Mitglieds des Sicherheitsteams zu versenden.

Die häufigste Ursache dafür ist zumeist ein Mail-Programm auf Ihrer Seite, das die Nachricht leicht verändert und dadurch die Signatur ungültig macht. Versichern Sie sich, dass Ihre Software keine MIME-Entschlüsselung oder -Verschlüsselung durchführt und auch keine Tabulator/Leerzeichen-Konvertierungen vornimmt.

Bekannte Übeltäter sind fetchmail (mit der Option mimedecode), formail (nur von procmail 3.14) und evolution.

12.3.3 Wie wird die Sicherheit in Debian gehandhabt?

Sobald das Sicherheitsteam auf einen Vorfall aufmerksam wird, überprüfen ihn ein oder mehrere Mitglieder und überlegen seinen Einfluss auf die Stable-Veröffentlichung von Debian (z.B. ob es verwundbar ist oder nicht). Wenn unser System verwundbar ist, arbeiten wir an einer Problembehebung. Der Paketbetreuer wird ebenfalls kontaktiert, wenn dieser nicht bereits

selbst das Sicherheits-Team kontaktiert hat. Schlussendlich wird die Behebung des Problems getestet und neue Pakete vorbereitet, die dann auf allen Stable-Architekturen übersetzt und anschließend hochgeladen wird. Nachdem das alles geschehen ist, wird eine Anweisung veröffentlicht.

12.3.4 Wieso spielen Sie mit einer alten Version des Pakets herum?

Die wichtigste Regel beim Erstellen eines neuen Pakets, das ein Sicherheitsproblem behebt, ist, so wenig Änderungen wie möglich vorzunehmen. Unsere Benutzer und Entwickler vertrauen auf das genaue Verhalten einer Veröffentlichung nach dessen Freigabe. Daher kann jede Änderung, die wir durchführen, möglicherweise das System von jemandem zerstören. Dies gilt insbesondere für Bibliotheken: Es muss darauf geachtet werden, dass sich das Anwendungs-Programm-Interface (API) oder Anwendungs-Binär-Interface (ABI) niemals ändert, egal wie klein die Änderung ist.

Dies bedeutet, dass das Umsteigen auf eine neue Version der Originalprogramms keine gute Lösung ist und stattdessen die relevanten Änderungen zurückportiert werden sollten. Üblicherweise sind die Betreuer des Originalprogramms, wenn notwendig, bereit zu helfen, falls das Debian-Sicherheitsteam nicht helfen kann.

In einigen Fällen ist es nicht möglich, eine Sicherheitsreparatur zurückzuportieren, zum Beispiel, wenn ein großer Teil des Quellcodes modifiziert oder neu geschrieben werden muss. Wenn dies passiert, kann es notwendig sein, auf eine neue Version des Originalprogramms umzusteigen, aber dies muss zuvor mit dem Sicherheits-Team koordiniert werden.

12.3.5 Was sind die Richtlinien für ein repariertes Paket, um auf security.debian.org zu erscheinen?

Sicherheitslücken in der Stable-Distribution garantieren, dass ein Paket zu security.debian.org hinzugefügt wird. Alles andere tut das nicht. Die Größe der Lücke ist hier nicht das wirkliche Problem. Üblicherweise bereitet das Sicherheitsteam die Pakete gemeinsam mit dem Paketbetreuer vor. Sofern jemand (ein Vertrauenswürdiger) das Problem analysiert, alle benötigten Pakete übersetzt und diese an das Sicherheitsteam übermittelt, dann können auch sehr kleine Sicherheitsreparaturen auf security.debian.org erscheinen. Lesen Sie dazu bitte auch unten weiter.

12.3.6 Die Versionsnummer für ein Paket zeigt an, dass ich immer noch eine verwundbare Version verwende!

Anstatt auf eine neue Veröffentlichung zu aktualisieren, portieren wir die Sicherheitsreparaturen auf die Version zurück, die mit der Stable-Veröffentlichung ausgeliefert wurde. Wir tun dies, um zu garantieren, dass eine Veröffentlichung so wenig wie möglich ändert, damit sich als Resultat des Sicherheitskorrekturen nichts ändert oder unerwartet Probleme verursacht werden. Sie können überprüfen, ob Sie eine sichere Version eines Paketes verwenden, indem

Sie die changelog-Datei des Paketes ansehen, oder seine exakte Versionsnummer mit der angezeigten Version in der Debian-Sicherheits-Ankündigung vergleichen.

12.3.7 Wie wird die Sicherheit für Testing und Unstable gehandhabt?

Die kurze Antwort ist: gar nicht. Testing und Unstable sind starken Änderungen unterworfen und das Sicherheitsteam hat nicht die Mittel, die benötigt würden, um diese entsprechend zu unterstützen. Wenn Sie einen sicheren (und stabilen) Server benötigen, wird Ihnen dringend empfohlen, bei Stable zu bleiben. Jedoch werden die Sicherheitssekretäre versuchen, die Probleme in Testing und Unstable zu beheben, nachdem sie in der Stable-Veröffentlichung behoben sind.

In einigen Fällen bekommt allerdings der Unstable-Zweig Sicherheitsreparaturen ziemlich schnell, da diese Reparaturen im Originalprogramm schneller verfügbar sind (dagegen müssen andere Versionen, wie die im Stable-Zweig, normalerweise zurückportiert werden).

12.3.8 Ich verwende eine ältere Version von Debian. Wird sie vom Debian-Sicherheitsteam unterstützt?

Nein. Unglücklicherweise kann das Sicherheitsteam von Debian nicht sowohl die Stable-Veröffentlichung (und inoffiziell auch Unstable) als auch andere, ältere Veröffentlichungen handhaben. Sie können allerdings Sicherheitsaktualisierungen für eine bestimmte Zeitspanne (normalerweise einige Monate) nach der Veröffentlichung einer neuen Debian-Distribution erwarten.

12.3.9 Warum gibt es keinen offiziellen Mirror von security.debian.org?

Der Zweck von security.debian.org ist es, Sicherheitsaktualisierungen möglichst schnell und einfach zur Verfügung zu stellen. Mirrors machen dies unnötig komplizierten und erzeugen nur Frustration, wenn sie nicht aktuell gehalten werden.

12.3.10 Ich habe DSA 100 und DSA 102 gesehen, doch wo ist DSA 101?

Verschiedene Distributoren (zumeist von GNU/Linux, aber auch von BSD-Derivaten) koordinieren Sicherheits-Ankündigung für verschiedene Vorfälle und haben vereinbart, einen bestimmten Zeitplan einzuhalten, so dass alle Distributoren in der Lage sind, eine Anweisung zur selben Zeit zu veröffentlichen. Dadurch soll verhindert werden, dass ein Anbieter diskriminiert wird, der mehr Zeit benötigt (z.B. falls der Hersteller längere Qualitätssicherungstests für die Pakete durchführt oder mehrere Architekturen bzw. Binär-Distributionen unterstützt). Unser eigenes Sicherheitsteam bereitet ebenfalls Anweisungen im Vorhinein vor. Es passiert immer wieder mal, dass andere Sicherheitsprobleme früher abgearbeitet werden müssen, als vorbereitete Gutachten veröffentlicht werden können, und daher wird temporär eine oder mehrere Anweisungen der Nummer nach ausgelassen.

12.3.11 Wie kann ich das Sicherheitsteam erreichen?

Sicherheitsinformationen können an security@debian.org (mailto:security@debian.org) geschickt werden, damit sie von allen Debian-Entwicklern gelesen werden. Wenn Sie sensible Informationen haben, benutzen Sie bitte team@security.debian.org (mailto:team@security.debian.org), die nur vom Sicherheitsteam gelesen wird. Wenn Sie es wünschen, kann die E-Mail auch mit dem Kontaktschlüssel von Debian-Security (Key-ID 0x363CCD95 (http://pgpkeys.pca.dfn.de:11371/pks/lookup?search=0x363CCD95op=vindex)) verschlüsselt werden.

12.3.12 Was ist der Unterschied zwischen security@debian.org und debian-security@lists.debian.org?

Wenn Sie eine Nachricht an security@debian.org schicken, wird diese an die Developer-Mailingliste geschickt (debian-private), die alle Entwickler von Debian abonniert haben. Nachrichten an diese Liste werden vertraulich behandelt (d.h. sie werden nicht auf einer öffentlichen Webseite archiviert). debian-security@lists.debian.org ist eine öffentliche Mail-Liste, offen für jeden, der Sie abonnieren (http://www.de.debian.org/MailingLists/) möchte, und es gibt hier (http://lists.debian.org/search.html) ein durchsuchbares Archiv.

12.3.13 Wie kann ich das Debian-Sicherheitsteam unterstützen?

- Indem Sie etwas zu diesem Dokument beitragen, FIXMEs bearbeiten oder neuen Inhalt beisteuern. Dokumentation ist wichtig und reduziert die Last durch Beantworten allgemeiner Fragen. Übersetzen dieses Dokuments in andere Sprachen ist auch ein großartiger Beitrag (Anm.d.Ü.: Fehler bereinigen in der Übersetzung auch).

- Indem Sie Pakete von Programmen erstellen, mit denen sich die Sicherheit eines Debian-Systems erhöhen oder prüfen lässt. Wenn Sie kein Entwickler sind, reichen Sie einen WNPP-Fehler (http://www.de.debian.org/devel/wnpp/) ein und fragen nach Software, die Sie für nützlich halten, die aber noch nicht zur Verfügung steht.

- Testen Sie Anwendungen in Debian oder helfen Sie Sicherheitslücken zu schließen. Teilen Sie Probleme security@debian.org mit. Die Arbeit anderer Projekte, wie das Linux Kernel Security Audit Project (http://kernel-audit.sourceforge.net/) oder das Linux Security Audit Project (http://www.lsap.org/) erhöhen auch die Sicherheit von Debian GNU/Linux, da Beiträge dort letzten Endes auch hier helfen.

Prüfen Sie bitte in jedem Fall ein Problem nach, bevor Sie es an security@debian.org melden. Wenn Sie Patches beifügen, beschleunigt das den Prozess natürlich. Leiten Sie nicht einfach Mails von Bugtraq weiter, da diese bereits empfangen wurden. Es ist aber eine gute Idee, zusätzliche Informationen zu schicken.

12.3.14 Aus wem setzt sich das Sicherheitsteam zusammen?

Das Debian-Sicherheitsteam besteht derzeit aus fünf Mitgliedern und zwei Sekretären. Das Sicherheitsteam bestimmt selbst, wen es neu ins Team aufnehmen will.

12.3.15 Prüft das Debian-Sicherheitsteam jedes Paket in Debian?

Nein, weder prüft das Sicherheitsteam jedes neue Paket noch gibt es einen automatischen (lintian) Test, um neue Pakete mit bösartigem Inhalt zu entdecken, da solche Dinge praktisch unmöglich automatisch zu erkennen sind. Paket-Verwalter sind jedoch voll und ganz verantwortlich für die Software, die sie in Debian einführen. Keine Software wird eingeführt, die nicht zuerst von einem autorisierten Entwickler signiert wurde. Die Entwickler sind dafür verantwortlich, die Sicherheit aller Pakete, die sie betreuen, zu analysieren.

12.3.16 Wie lange braucht Debian, um die Angriffs-Möglichkeit XXXX zu reparieren?

Das Sicherheitsteam von Debian arbeitet schnell, um Anweisungen zu verschicken und korrigierte Pakete für den Stable-Zweig zu erstellen, sobald eine Sicherheitslücke entdeckt wurde. Ein Bericht, der auf der Mailingliste von debian-security veröffentlicht wurde (`http://lists.debian.org/debian-security/2001/debian-security-200112/msg00257.html`), zeigt, dass das Debian-Sicherheitsteam im Jahr 2001 durchschnittlich 35 Tage gebraucht hat, um Sicherheitslücken auszubessern. Allerdings wurden über 50% der Verwundbarkeiten innerhalb von zehn Tagen beseitigt und über 15% wurden am *gleichen Tag* repariert, an dem die Anweisung veröffentlicht wurde.

Oft vergessen Leute, die diese Frage stellen, dass:

- DSAs nicht verschickt werden bis:
  - Pakete für *alle* von Debian unterstützten Architekturen verfügbar sind (dies braucht etwas Zeit, wenn es sich um Pakete handelt, die Teil des Systemkerns sind, besonders wenn man die Anzahl der in der stabilen Veröffentlichung unterstützten Architekturen berücksichtigt).
  - Neue Pakete gründlich getestet werden, um sicher zu stellen, dass sich keine neuen Fehler eingeschlichen haben.
- Pakete möglicherweise schon verfügbar sind, bevor das DSA verschickt wird (in der Incoming-Warteschlange oder auf den Mirror-Servern).
- Debian ein Projekt auf freiwilligen Basis ist.
- Es eine “keine Garantie”-Klausel gibt, die Teil der Lizenz ist, der Debian unterliegt.

Wenn Sie eine tiefergehende Analyse wünschen, wie lange das Sicherheitsteam an Sicherheitslücken arbeitet, sollten Sie wissen, dass neue DSAs (vergleichen Sie 'Debian-Sicherheits-Ankündigungen' auf Seite 138) auf der Sicherheits-Webseite (`http://security.de.debian.org`) veröffentlicht werden. Daneben finden sich dort auch die Metadaten, die verwendetet werden, um die DSAs zu erstellen, und Links zur Datenbank mit den Sicherheitslücken. Sie können die Quellen vom Webserver herunterladen (aus dem CVS (`http://cvs.debian.org`)) oder die HTML-Seiten benutzen, um zu bestimmen, wie lange Debian braucht, um Verwundbarkeiten auszubessern, und um diese Daten mit öffentlichen Datenbanken zu vergleichen.

Anhang A

Der Abhärtungsprozess Schritt für Schritt

Eine Anleitung, die Schritt für Schritt darstellt, wie ein Debian 2.2 GNU/Linux-System nach der Installation abgehärtet wird, ist unten aufgeführt. Das ist nur eine denkbare Herangehensweise von einem solchen Vorgang. Sie ist am Absichern von Netzwerkdiensten orientiert und stellt den gesamten Anlauf der Konfiguration vor. Vergleichen Sie auch 'Checkliste der Konfiguration' auf Seite 227.

- Installieren Sie das System. Beachten Sie dabei die Informationen dieses HOWTOs bezüglich der Partitionierung. Nach der Basis-Installation nehmen Sie eine angepasste Installation vor. Wählen Sie keine Task-Pakete aus. Aktivieren Sie shadow passwords.
- Entfernen Sie mit `dselect` alle nicht benötigten aber ausgewählten Pakete, bevor Sie [I]nstallation wählen. Behalten Sie nur die absolut notwendige Software auf dem System.
- Aktualisieren Sie die ganze Software mit den aktuellen Paketen von security.debian.org, wie bereits unter 'Ausführen von Sicherheitsupdates' auf Seite 48 beschrieben.
- Implementieren Sie die in dieser Anleitung vorgeschlagenen Maßnahmen zu User-Quotas, Ausgestaltung des Logins und `Lilo`.
- Machen Sie sich eine Liste von allen Diensten, die derzeit auf Ihrem System laufen. Versuchen Sie dazu Folgendes:

  ```
  $ ps aux
  $ netstat -pn -l -A inet
  # /usr/sbin/lsof -i | grep LISTEN
  ```

  Damit das dritte Kommando funktioniert, werden Sie `lsof-2.2` installieren müssen (und es als Root laufen lassen). Beachten Sie, dass `lsof` das Wort LISTEN passend zu Ihrer Lokalisation übersetzen kann.

- Um einen unnötigen Dienst zu entfernen, stellen Sie zunächst fest, wie er gestartet wird, und welches Paket ihn zur Verfügung stellt. Sie können dies ganz einfach machen, indem Sie das Programm prüfen, das auf dem Socket lauscht. Das nachfolgende Shell-Skript, das die Programme `lsof` und `dpkg` verwendet, macht genau das:

```
#!/bin/sh
# FIXME: this is quick and dirty; replace with a more robust script sni
for i in ‘sudo lsof -i | grep LISTEN | cut -d " " -f 1 |sort -u‘ ; do
  pack=‘dpkg -S $i |grep bin |cut -f 1 -d : | uniq‘
  echo "Service $i is installed by $pack";
  init=‘dpkg -L $pack |grep init.d/ ‘
  if [ ! -z "$init" ]; then
    echo "and is run by $init"
  fi
done
```

- Wenn Sie einen unerwünschten Dienst finden, entfernen Sie das Paket (mit `dpkg -purge`). Oder benutzen Sie `update-rc.d` (siehe ‘Daemons abschalten’ auf Seite 40), um ihn aus dem Start-Prozess zu entfernen.

- Überprüfen Sie bei inetd-Diensten (werden durch den Superdaemon gestartet), welche Dienste in `/etc/inetd.conf` aktiviert sind. Verwenden Sie dazu Folgendes:

```
$ grep -v "^#" /etc/inetd.conf | sort -u
```

  Deaktivieren Sie dann diejenigen Dienste, die Sie nicht benötigen, indem Sie die Zeile in `/etc/inetd.conf` auskommentieren, das Paket entfernen, oder indem Sie `update-inetd` benutzen.

- Wenn Sie Dienste eingehüllt haben (und `/usr/sbin/tcpd` benutzen) prüfen Sie, ob die Dateien `/etc/hosts.allow` und `/etc/hosts.deny` passend zu Ihren Richtlinien für die Dienste konfiguriert sind.

- Wenn der Server mehr als eine externe Schnittstelle benutzt, sollten Sie vielleicht Dienste darauf beschränken, auf bestimmten Schnittstellen zu lauschen. Ob das möglich ist, hängt aber von den Diensten ab. Wenn Sie zum Beispiel internen FTP-Zugriff erlauben wollen, lassen Sie den FTP-Daemon nur auf der internen Schnittstelle lauschen, nicht auf allen (d.h. 0.0.0.0:21).

- Booten Sie die Maschine neu, oder wechseln Sie in den Single-User-Modus und zurück in den Multi-User-Modus mit:

```
$ init 1
(....)
$ init 2
```

- Prüfen Sie die nun angebotenen Dienste und wiederholen Sie gegebenenfalls die letzten Schritte.
- Installieren Sie jetzt die benötigten Dienste, falls es noch nicht geschehen ist, und konfigurieren Sie sie passend.
- Prüfen Sie mit folgendem Shell-Befehl, unter welchem Nutzer die verfügbaren Dienste laufen:

```
$ for i in ‘/usr/sbin/lsof -i |grep LISTEN |cut -d " " -f 1 |sort -u‘
> do user=‘ps ef |grep $i |grep -v grep |cut -f 1 -d " "‘ ; \
> echo "Service $i is running as user $user"; done
```

  Überlegen Sie, ob Sie diese Dienste einem bestimmten Benutzer oder Gruppe zuweisen wollen und sie vielleicht auch in einer chroot-Umgebung einsperren wollen, um die Sicherheit zu erhöhen. Sie können dies tun, indem Sie die `/etc/init.d`-Skripte ändern, die den Dienst starten. Die meisten Dienste benutzen unter Debian `start-stop-daemon`, der die dafür Optionen (`--change-uid` und `--chroot`) zur Verfügung stellen. Ein paar warnende Worte zum Einsperren in eine `chroot`-Umgebung: Sie müssen alle Dateien, die durch das Paket des Dienstes installiert wurden (verwenden Sie dpkg -L), und alle Pakete, von denen es abhängt, in die `Chroot`-Umgebung legen. Informationen, wie das Programm `ssh` in eine `chroot`-Umgebung eingesperrt wird, finden Sie unter ‘`Chroot`-Umgebung für `SSH`’ auf Seite 249.
- Wiederholen Sie die Schritte oben um zu prüfen, ob nur die gewünschten Dienste laufen, und ob sie unter der gewünschten Nutzer/Gruppen-Kombination laufen.
- Testen Sie die installierten Dienste, um festzustellen, ob sie wie erwartet arbeiten.
- Überprüfen Sie das System, indem Sie einen Scanner zur Abschätzung der Verwundbarkeit (zum Beispiel `nessus`) benutzen, um Angriffsmöglichkeiten (Fehlkonfigurationen, alte oder nicht benötigte Dienste) zu finden.
- Installieren Sie Instrumente zur Entdeckung von Eindringlingen in Netzwerk und Hosts (wie `snort` und `logcheck`).
- Wiederholen Sie den Netzwerk-Scan und prüfen Sie, ob das System zur Erkennung von Eindringlingen funktioniert.

Die richtig Paranoiden überlegen sich auch das Folgende:

- Fügen Sie dem System Firewall-Fähigkeiten hinzu, die eingehende Verbindungen nur zu angebotenen Diensten erlauben und ungenehmigte ausgehende Verbindungen verhindern.
- Überprüfen Sie erneut die Installation auf Angriffspunkte mit einem Netzwerk-Scanner.

- Prüfen Sie ausgehende Verbindungen vom System zu Hosts außerhalb mit einem Netzwerk-Scanner, um sicherzustellen, dass ungewollte Verbindungen keinen Weg nach draußen finden.

FIXME: this procedure considers service hardening but not system hardening at the user level, include information regarding checking user permissions, SETUID files and freezing changes in the system using the ext2 file system.

Anhang B

Checkliste der Konfiguration

Dieser Anhang wiederholt kurz Punkte aus anderen Abschnitten dieser Anleitung in einem verdichteten Checklisten-Format. Er ist als schnelle Zusammenfassung für Leute gedacht, die bereits diese Anleitung gelesen haben. Es gibt auch andere gute Checklisten, zum Beispiel Kurt Seifrieds Securing Linux Step by Step (`http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html`) und CERT's Unix Security Checklist (`http://www.cert.org/tech_tips/usc20_full.html`).

FIXME: This is based on v1.4 of the manual and might need to be updated.

- Schränken Sie physischen Zugriff und Boot-Fähigkeiten ein.
  - Setzen Sie ein BIOS-Passwort.
  - Schalten Sie das Booten von Diskette, CD-ROM, ... ab.
  - Setzen Sie ein LILO- bzw. GRUB-Passwort (`/etc/lilo.conf` bzw. `/boot/grub/menu.lst`); stellen Sie sicher, dass die LILO- oder GRUB-Konfigurationen nicht einsehbar sind.
  - Verhindern Sie die Hintertür des Bootens von Diskette durch den MBR, indem Sie den MBR überschreiben (vielleicht nicht?)
- Partitionierung:
  - Legen Sie Daten, die von Nutzern geschrieben wurden, Daten, die nicht zum System gehören, und sich ständig ändernde Laufzeitdaten auf eigenen, getrennten Partitionen ab.
  - Setzen Sie die Mount-Optionen `nosuid,noexec,nodev` in `/etc/fstab` bei ext2-Partitionen, wie zum Beispiel `/tmp`.
- Passwort-Hygiene und Login-Sicherheit:
  - Wählen Sie ein gutes Root-Passwort.
  - Benutzen Sie Shadow- und MD5-Passwörter.

  - Installieren und benutzen Sie PAM:
    * Fügen Sie die Unterstützung von PAM-MD5 hinzu, und stellen Sie sicher (allgemein gesprochen), dass die Einträge in den `/etc/pam.d/`-Dateien, die Zugriff auf die Maschine gewähren, das zweite Feld in der pam.d-Datei auf `requisite` oder `required` gesetzt haben.
    * Ändern Sie `/etc/pam.d/login`, so dass nur lokale Root-Logins erlaubt werden.
    * Markieren Sie außerdem autorisierte ttys in `/etc/security/access.conf` und richten Sie diese Datei überhaupt so ein, dass Root-Logins so weit wie möglich eingeschränkt werden.
    * Fügen Sie pam_limits.so hinzu, wenn Sie Einschränkungen pro Benutzer vornehmen wollen.
    * Ändern Sie `/etc/pam.d/passwd`: Erhöhen Sie die minimale Länge von Passwörtern (vielleicht sechs Zeichen) und schalten Sie MD5 ein.
    * Wenn Sie es wünschen, fügen Sie `/etc/group` die Gruppe wheel hinzu; fügen Sie `/etc/pam.d/su` pam_wheel.so group=wheel hinzu.
    * Für angepasste Kontrollen der einzelnen Nutzer nehmen Sie pam_listfile.so-Einträge an den passenden Stellen vor.
    * Erstellen Sie eine Datei `/etc/pam.d/other` und setzen Sie sie mit strenger Sicherheit auf.
  - Setzen Sie in `/etc/security/limits.conf` Schranken (beachten Sie, dass `/etc/limits` nicht benutzt wird, wenn Sie PAM verwenden).
  - Nehmen Sie Einschränkungen in `/etc/login.defs` vor; wenn Sie MD5 oder PAM einschalten, machen Sie auch hier ebenfalls die gleichbedeutenden Änderungen.
  - Schalten Sie FTP-Zugriff von Root in `/etc/ftpusers` ab.
  - Schalten Sie Root-Logins übers Netzwerk ab; benutzen Sie `su(1)` oder `sudo(1)` (denken Sie die Installation von `sudo` nach).
  - Benutzen Sie PAM, um zusätzliche Auflagen auf Logins zu ermöglichen.

- Andere lokale Sicherheitsangelegenheiten:
  - Kernel-Tweaks (siehe 'Konfiguration der Netzwerkfähigkeiten des Kernels' auf Seite 87).
  - Kernel-Patches (siehe 'Den Kernel patchen' auf Seite 78).
  - Schränken Sie die Zugriffsrechte auf Log-Dateien (`/var/log/{last,fail}log`, Apache-Logs) ein.
  - Stellen Sie sicher, dass in `/etc/checksecurity.conf` SETUID-Checks eingeschaltet sind.
  - Überlegen Sie sich, an Log-Dateien nur anhängen zu lassen (append-only) und Konfigurationsdateien unveränderbar (immutable) zu machen, indem Sie chattr benutzen (nur ext2-Dateisystem).
  - Setzen Sie eine Integritätsprüfung des Dateisystems auf (siehe 'Prüfung der Integrität des Dateisystems' auf Seite 85). Installieren Sie `debsums`.

  - Überlegen Sie, locate durch slocate zu ersetzen.
  - Alles auf einem lokalen Drucker mitloggen?
  - Brennen Sie Ihre Konfiguration auf eine bootbare CD und booten Sie hiervon?
  - Abschalten von Kernel-Modulen?

- Einschränkung des Netzwerkzugriffs:

  - Installieren und konfigurieren Sie `ssh` (Vorschlag: PermitRootLogin No in `/etc/ssh/sshd_config`, PermitEmptyPasswords No; beachten Sie auch die anderen Vorschläge im Text).
  - Überlegen Sie, ob Sie `in.telnetd` abschalten oder entfernen.
  - Deaktivieren Sie ganz allgemein alle überflüssigen Dienste in `/etc/inetd.conf`. Benutzen Sie dazu `update-inetd -disable` (oder Sie schalten `inetd` ganz ab oder benutzen einen Ersatz wie `xinetd` oder `rlinetd`).
  - Schalten Sie andere überflüssige Netzwerkdienste ab; mail, ftp, DNS, www, usw. sollten nicht laufen, wenn Sie sie nicht brauchen und nicht regelmäßig überwachen.
  - Installieren Sie von den Diensten, die Sie brauchen, nicht einfach das weit verbreitetste Programm, sondern schauen Sie nach sichereren Versionen, die Debian liefert (oder aus anderen Quellen). Was auch immer Sie schließlich benutzen: Stellen Sie sicher, dass Sie die Risiken verstanden haben.
  - Setzen Sie `Chroot`-Gefängnisse für auswärtige Nutzer und Daemonen auf.
  - Konfigurieren Sie die Firewall und die tcp-Wrapper (d.h. `hosts_access(5)`); beachten Sie den Trick für `/etc/hosts.deny` im Text.
  - Wenn Sie FTP laufen lassen, setzen Sie den ftpd-Server so auf, dass er immer in einer `chroot`-Umgebung im Home-Verzeichnis des Nutzers läuft.
  - Wenn Sie X laufen lassen, schalten Sie xhost-Authentifizierung ab und benutzen Sie stattdessen `ssh`. Oder noch besser: Deaktivieren Sie die Weiterleitung von X komplett, falls das möglich ist (fügen Sie -nolisten tcp zu der X-Kommando-Zeile hinzu und schalten Sie XDMCP in `/etc/X11/xdm/xdm-config` ab, indem Sie den requestPort auf 0 setzen).
  - Schalten Sie Zugriff von außerhalb auf den Drucker ab.
  - Tunneln Sie alle IMAP- oder POP-Sitzungen durch SSL oder `ssh`. Installieren Sie stunnel, wenn Sie diesen Dienst anderen Mail-Nutzern anbieten wollen.
  - Setzen Sie einen Log-Host auf, und konfigurieren Sie andere Maschinen, ihre Logs an diesen Host zu senden (`/etc/syslog.conf`)
  - Sichern Sie BIND, Sendmail und andere komplexe Daemonen ab (starten Sie sie in einer `chroot`-Umgebung und als nicht-Root Pseudonutzer).
  - Installieren Sie snort oder ein ähnliches Überwachungswerkzeug.
  - Verzichten Sie, falls möglich, auf NIS und RPC (Abschalten von portmap).

- Angelegenheiten mit Richtlinien:

  - Klären Sie die Nutzer über das Wie und Warum Ihrer Richtlinien auf. Wenn Sie etwas verboten haben, das auf anderen Systemen normalerweise verfügbar ist, stellen Sie Dokumentation bereit, die erklärt, wie man die gleichen Resultate erreicht, indem man andere, sichere Mittel anwendet.
  - Verbieten Sie die Nutzung von Protokollen, die Klartext-Passwörter benutzen (`telnet`, `rsh` und Freunde, ftp, imap, pop, http, ...).
  - Verbieten Sie Programme, die SVGAlib benutzen.
  - Benutzen Sie Disk-Quotas.

- Bleiben Sie über Sicherheitsangelegenheiten informiert:
  - Abonnieren Sie sicherheitsrelevante Mailinglisten.
  - Richten Sie Sicherheitsaktualisierungen für `apt` ein – fügen Sie `/etc/apt/sources.list` einen Eintrag (oder Einträge) für http://security.debian.org/debian-security hinzu.
  - Vergessen Sie auch nicht, regelmäßig `apt-get update ; apt-get upgrade` (vielleicht als `Cron`-Job?) laufen zu lassen, wie unter 'Ausführen von Sicherheitsupdates' auf Seite 48 beschrieben.

Anhang C

Aufsetzen eines autonomen IDS

Sie können sehr leicht eine Debian-Box als eigenständiges Eindringlings-Erkennungs-System (Intrusion Detection System, IDS) aufsetzen, indem Sie `snort` benutzen.

Ein kleiner Leitfaden:

- Installieren Sie ein Debian Basis-System ohne zusätzliche Pakete.
- Laden Sie notwendige Pakete (siehe die Liste der installierten Pakete weiter unten) manuell herunter und installieren Sie sie (mit dpkg).
- Laden Sie ACID (Analysis Console for Intrusion Databases, Konsole zur Analyse für Eindringling-Datenbanken) herunter und installieren Sie es.

ACID wird derzeit für Debian im Paket `acidlab` geliefert. Es stellt eine graphische WWW-Schnittstelle zur Ausgabe von Snort zur Verfügung. Es kann von `http://www.cert.org/kb/acid/`, `http://acidlab.sourceforge.net` oder `http://www.andrew.cmu.edu/~rdanyliw/snort/` heruntergeladen werden. Sie sollten vielleicht auch das Snort Statistics HOWTO (`http://www.tldp.org/HOWTO/Snort-Statistics-HOWTO/index.html`) lesen.

Dieses System sollte mit wenigstens zwei Netzwerk-Schnittstellen ausgestattet sein: Eine verbunden mit einem Verwaltungs-LAN (um die Resultate abzufragen und das System zu verwalten), und eine ohne IP-Adresse, das an mit dem zu beobachtenden Abschnitt des Netzwerks verbunden ist.

Sie können nicht die Standard-Debian-Datei `/etc/network/interfaces` benutzen, die normalerweise dazu verwendet wird, Netzwerkkarten zu konfigurieren, da `ifup` und `ifdown` eine IP-Adresse erwarten. Benutzen stattdessen Sie einfach `ifconfig eth0 up`.

Abgesehen von der Standard-Installation benötigt `acidlab` unter anderem `php4` und `apache`. Laden Sie die folgenden Pakete herunter (beachten Sie: Die Versionen können abhängig von der verwendeten Debian-Distribution variieren. Diese sind aus Debian *Woody* vom September 2001):

```
ACID-0.9.5b9.tar.gz
adduser_3.39_all.deb
apache-common_1.3.20-1_i386.deb
apache_1.3.20-1_i386.deb
debconf_0.9.77_all.deb
dialog_0.9a-20010527-1_i386.deb
fileutils_4.1-2_i386.deb
klogd_1.4.1-2_i386.deb
libbz2-1.0_1.0.1-10_i386.deb
libc6_2.2.3-6_i386.deb
libdb2_2.7.7-8_i386.deb
libdbd-mysql-perl_1.2216-2_i386.deb
libdbi-perl_1.18-1_i386.deb
libexpat1_1.95.1-5_i386.deb
libgdbmg1_1.7.3-27_i386.deb
libmm11_1.1.3-4_i386.deb
libmysqlclient10_3.23.39-3_i386.deb
libncurses5_5.2.20010318-2_i386.deb
libpcap0_0.6.2-1_i386.deb
libpcre3_3.4-1_i386.deb
libreadline4_4.2-3_i386.deb
libstdc++2.10-glibc2.2_2.95.4-0.010703_i386.deb
logrotate_3.5.4-2_i386.deb
mime-support_3.11-1_all.deb
mysql-client_3.23.39-3_i386.deb
mysql-common_3.23.39-3.1_all.deb
mysql-server_3.23.39-3_i386.deb
perl-base_5.6.1-5_i386.deb
perl-modules_5.6.1-5_all.deb
perl_5.6.1-5_i386.deb
php4-mysql_4.0.6-4_i386.deb
php4_4.0.6-1_i386.deb
php4_4.0.6-4_i386.deb
snort_1.7-9_i386.deb
sysklogd_1.4.1-2_i386.deb
zlib1g_1.1.3-15_i386.deb
```

Installierte Pakete (dpkg -l):

```
ii  adduser        3.39
ii  ae             962-26
ii  apache         1.3.20-1
ii  apache-common  1.3.20-1
ii  apt            0.3.19
ii  base-config    0.33.2
```

```
ii  base-files     2.2.0
ii  base-passwd    3.1.10
ii  bash           2.03-6
ii  bsdutils       2.10f-5.1
ii  console-data   1999.08.29-11.
ii  console-tools  0.2.3-10.3
ii  console-tools- 0.2.3-10.3
ii  cron           3.0pl1-57.2
ii  debconf        0.9.77
ii  debianutils    1.13.3
ii  dialog         0.9a-20010527-
ii  diff           2.7-21
ii  dpkg           1.6.15
ii  e2fsprogs      1.18-3.0
ii  elvis-tiny     1.4-11
ii  fbset          2.1-6
ii  fdflush        1.0.1-5
ii  fdutils        5.3-3
ii  fileutils      4.1-2
ii  findutils      4.1-40
ii  ftp            0.10-3.1
ii  gettext-base   0.10.35-13
ii  grep           2.4.2-1
ii  gzip           1.2.4-33
ii  hostname       2.07
ii  isapnptools    1.21-2
ii  joe            2.8-15.2
ii  klogd          1.4.1-2
ii  ldso           1.9.11-9
ii  libbz2-1.0     1.0.1-10
ii  libc6          2.2.3-6
ii  libdb2         2.7.7-8
ii  libdbd-mysql-p 1.2216-2
ii  libdbi-perl    1.18-1
ii  libexpat1      1.95.1-5
ii  libgdbmg1      1.7.3-27
ii  libmm11        1.1.3-4
ii  libmysqlclient 3.23.39-3
ii  libncurses5    5.2.20010318-2
ii  libnewt0       0.50-7
ii  libpam-modules 0.72-9
ii  libpam-runtime 0.72-9
ii  libpam0g       0.72-9
ii  libpcap0       0.6.2-1
ii  libpcre3       3.4-1
ii  libpopt0       1.4-1.1
```

```
ii  libreadline4   4.2-3
ii  libssl09       0.9.4-5
ii  libstdc++2.10  2.95.2-13
ii  libstdc++2.10- 2.95.4-0.01070
ii  libwrap0       7.6-4
ii  lilo           21.4.3-2
ii  locales        2.1.3-18
ii  login          19990827-20
ii  makedev        2.3.1-46.2
ii  mawk           1.3.3-5
ii  mbr            1.1.2-1
ii  mime-support   3.11-1
ii  modutils       2.3.11-13.1
ii  mount          2.10f-5.1
ii  mysql-client   3.23.39-3
ii  mysql-common   3.23.39-3.1
ii  mysql-server   3.23.39-3
ii  ncurses-base   5.0-6.0potato1
ii  ncurses-bin    5.0-6.0potato1
ii  netbase        3.18-4
ii  passwd         19990827-20
ii  pciutils       2.1.2-2
ii  perl           5.6.1-5
ii  perl-base      5.6.1-5
ii  perl-modules   5.6.1-5
ii  php4           4.0.6-4
ii  php4-mysql     4.0.6-4
ii  ppp            2.3.11-1.4
ii  pppconfig      2.0.5
ii  procps         2.0.6-5
ii  psmisc         19-2
ii  pump           0.7.3-2
ii  sed            3.02-5
ii  setserial      2.17-16
ii  shellutils     2.0-7
ii  slang1         1.3.9-1
ii  snort          1.7-9
ii  ssh            1.2.3-9.3
ii  sysklogd       1.4.1-2
ii  syslinux       1.48-2
ii  sysvinit       2.78-4
ii  tar            1.13.17-2
ii  tasksel        1.0-10
ii  tcpd           7.6-4
ii  telnet         0.16-4potato.1
ii  textutils      2.0-2
```

```
ii  update          2.11-1
ii  util-linux      2.10f-5.1
ii  zlib1g          1.1.3-15
```

Anhang D

Aufsetzen einer überbrückenden Firewall (bridge Firewall)

Diese Informationen trug Francois Bayart bei, um Benutzern zu helfen, eine Linux Bridge/Firewall mit 2.4.x Kernel und `iptables` aufzusetzen. Ein Kernelpatch wird nicht mehr benötigt, da der Code Standardinhalt der Linux-Kernel-Distribution wurde.

Um die notwendigen Einstellungen im Kernel vorzunehmen, rufen Sie `make menuconfig` oder `make xconfig` auf. Aktivieren Sie im Abschnitt *Networking options* folgende Optionen:

```
[*] Network packet filtering (replaces ipchains)
[ ]   Network packet filtering debugging (NEW)
<*> 802.1d Ethernet Bridging
[*]   netfilter (firewalling) support (NEW)
```

Passen Sie auf, dass Sie dieses hier deaktiviert haben, wenn Sie Firewall-Regeln anwenden wollen. Anderenfalls wird iptables nicht funktioniert.

```
[ ]   Network packet filtering debugging (NEW)
```

Anschließend müssen Sie die korrekten Optionen im Abschnitt *IP: Netfilter Configuration* setzen. Dann kompilieren und installieren Sie den Kernel. Wenn Sie dies auf die *Debian-Art* machen wollen, installieren Sie `kernel-package` und benutzen Sie `make-kpkg` um ein maßgeschneidertes Debian-Kernelpaket zu erstellen, das Sie mit dpkg auf Ihrem Server installieren können. Sobald der neue Kernel kompiliert und installiert ist, müssen Sie das Paket `bridge-utils` installieren.

Wenn Sie diesen Schritt abgeschlossen haben, können Sie die Konfiguration Ihrer Bridge fertigstellen. Im nächsten Abschnitt werden Ihnen zwei verschiedene mögliche Konfigurationen einer Bridge vorgestellt. Beide sind mit einer Übersicht eines hypothetischen Netzwerks und den notwendigen Befehlen versehen.

D.1 Eine Bridge mit NAT- und Firewall-Fähigkeiten

Die erste Konfigurationsmöglichkeit benutzt die Bridge als Firewall mit Network Address Translation (NAT, Übersetzung der Netzwerkadressen), die einen Server und interne LAN-Clienten schützt. Unten wird eine Darstellung der Anordnung des Netzwerks gezeigt:

```
Internet ---- router ( 62.3.3.25 ) ---- bridge (62.3.3.26 gw 62.3.3.25 / 192.
                                          |
                                          |
                                          |---- WWW Server (62.3.3.27 gw 62.3
                                          |
                                          |
                                        LAN --- Zipowz (192.168.0.2 gw 192.1
```

Die folgenden Befehle zeigen, wie diese Bridge konfiguriert werden kann:

```
# So wird die Schnittstelle br0 erstellt:
/usr/sbin/brctl addbr br0

# Hinzufügen der Ethernet-Schnittstelle, die die Bridge benutzen
# soll
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1

# Starten der Ethernet-Schnittstelle
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig eth1 0.0.0.0

# Konfigurieren der Ethernet-Bridge
# Die Bridge wird korrekt und unsichtbar (transparente Firewall) sein.
# In einem traceroute ist sie versteckt, und Sie behalten Ihr echtes
# Gateway auf Ihren anderen Computern. Jetzt können Sie ein Gateway
# auf Ihrer Bridge konfigurieren und es auf Ihren anderen Computern als
# neues Gateway einsetzen

/sbin/ifconfig br0 62.3.3.26 netmask 255.255.255.248 broadcast 62.3.3.32

# Ich habe diese internen IPs für mein NAT benutzt
ip addr add 192.168.0.1/24 dev br0
/sbin/route add default gw 62.3.3.25
```

D.2 Eine Bridge mit Firewall-Fähigkeiten

Eine zweite denkbare Konfiguration ist ein System, das als transparente Firewall für ein LAN mit einer öffentlichen IP-Adresse aufgesetzt ist.

```
Internet ---- router (62.3.3.25) ---- bridge (62.3.3.26)
                                          |
                                          |
                                          |---- WWW Server (62.3.3.28 gw 62.3.3
                                          |
                                          |
                                          |---- Mail Server (62.3.3.27 gw 62.3.
```

Die folgenden Kommandos zeigen, wie diese Bridge konfiguriert werden kann:

```
# So wird die Schnittstelle br0 erstellt:
/usr/sbin/brctl addbr br0

# Hinzufügen der Ethernet-Schnittstelle, die die Bridge benutzen
# soll
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1

# Starten der Schnittstelle
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig eth1 0.0.0.0

# Konfigurieren der Ethernet-Bridge
# Die Bridge wird korrekt und unsichtbar (transparente Firewall) sein.
# In einem traceroute ist sie versteckt, und Sie behalten Ihr echtes
# Gateway auf Ihren anderen Computern. Jetzt können Sie ein Gateway
# auf Ihrer Bridge konfigurieren und es auf Ihren anderen Computern als
# neues Gateway einsetzen

/sbin/ifconfig br0 62.3.3.26 netmask 255.255.255.248 broadcast 62.3.3.32
```

Wenn Sie mit `traceroute` die Route des Linux-Mail-Servers verfolgen, sehen Sie die Bridge nicht. Wenn Sie mit `ssh` auf die Bridge zugreifen wollen, müssen Sie ein Gateway haben oder erst auf einen anderen Server wie den "Mail Server" zugreifen, um dann über die interne Netzwerkkarte auf die Bridge zuzugreifen.

D.3 Grundlegende Iptables-Regeln

Dies ist ein Beispiel für grundlegende Regeln, die für beide Einstellungen benutzt werden können:

```
iptables -F FORWARD
iptables -P FORWARD DROP
```

```
iptables -A FORWARD -s 0.0.0.0/0.0.0.0 -d 0.0.0.0/0.0.0.0 -m state --state IN
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Zwei lustige Regeln, aber nicht bei klassischen Iptables. Sorry ...
# Limit ICMP
# iptables -A FORWARD -p icmp -m limit --limit 4/s -j ACCEPT
# Übereinstimmende Strings, eine gute, einfache Methode, um Viren sehr
# schnell abzublocken
# iptables -I FORWARD -j DROP -p tcp -s 0.0.0.0/0 -m string --string "cmd.exe

# Abblocken aller MySQL-Verbindungen, nur um ganz sicher zu gehen
iptables -A FORWARD -p tcp -s 0/0 -d 62.3.3.0/24 --dport 3306 -j DROP

# Regeln für den Linux Mail Server
#

# Erlaube FTP-DATA (20), FTP (21), SSH (22)
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.27/32 --dport 20:22 -j ACCE

# Dem Mail-Server erlauben, sich mit der Außenwelt zu verbinden
# Beachten Sie: Dies ist *nicht* für die vorherigen Verbindungen
# notwendig (erinnern Sie sich: stateful filtering) und könnte entfernt
# werden:
iptables -A FORWARD -p tcp -s 62.3.3.27/32 -d 0/0 -j ACCEPT

# Regeln für den WWW-Server
#

# Erlaube HTTP ( 80 ) Verbindungen mit dem WWW-Server
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.28/32 --dport 80 -j ACCEPT

# Erlaube HTTPS ( 443 ) Verbindungen mit dem WWW-Server
iptables -A FORWARD -p tcp -s 0.0.0.0/0 -d 62.3.3.28/32 --dport 443 -j ACCEPT

# Dem WWW-Server erlauben, sich mit der Außenwelt zu verbinden
# Beachten Sie: Dies ist *nicht* für die vorherigen Verbindungen
# notwendig (erinnern Sie sich: stateful filtering) und könnte entfernt
# werden:
iptables -A FORWARD -p tcp -s 62.3.3.28/32 -d 0/0 -j ACCEPT
```

Anhang E

Beispielskript, um die Standard-Installation von Bind zu ändern

Dieses Skript automatisiert die Prozedur, die Standard-Installation des Name-Servers `bind` zu ändern, so dass er *nicht* als Superuser läuft. Es wird auch den Nutzer und die Gruppe für den Name-Server erstellen. Benutzen Sie es äußerst vorsichtig, da es nicht ausreichend getestet wurde.

```
#!/bin/sh
# Change the default Debian bind configuration to have it run
# with a non-root user and group.
#
# WARN: This script has not been tested thoroughly, please
# verify the changes made to the INITD script

# (c) 2002 Javier Fernández-Sanguino Peña
#
#    This program is free software; you can redistribute it and/or modify
#    it under the terms of the GNU General Public License as published by
#    the Free Software Foundation; either version 1, or (at your option)
#    any later version.
#
#    This program is distributed in the hope that it will be useful,
#    but WITHOUT ANY WARRANTY; without even the implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#    GNU General Public License for more details.
#
#     Please see the file 'COPYING' for the complete copyright notice.
#
```

```
restore() {
# Just in case, restore the system if the changes fail
  echo "WARN: Restoring to the previous setup since I'm unable to properly
  echo "WARN: Please check the $INITDERR script."
  mv $INITD $INITDERR
  cp $INITDBAK $INITD
}


USER=named
GROUP=named
INITD=/etc/init.d/bind
INITDBAK=$INITD.preuserchange
INITDERR=$INITD.changeerror
START="start-stop-daemon --start --quiet --exec /usr/sbin/named -- -g $GROU
AWKS="awk ' /start-stop-daemon --start/ { print \"$START\"; noprint = 1; };

[ `id -u` -ne 0 ] && {
  echo "This program must be run by the root user"
  exit 1
}

RUNUSER=`ps eo user,fname |grep named |cut -f 1 -d " "`

if [ "$RUNUSER" = "$USER" ]
then
  echo "WARN: The name server running daemon is already running as $USER"
  echo "ERR:  This script will not many any changes to your setup."
  exit 1
fi
if [ ! -f $INITD ]
then
  echo "ERR:  This system does not have $INITD (which this script tries to
  RUNNING=`ps eo fname |grep named`
  [ -z "$RUNNING" ] && \
    echo "ERR:  In fact the name server daemon is not even running (is it i
  echo "ERR:  No changes will be made to your system"
  exit 1
fi

# Check if named group exists
if [ -z "`grep $GROUP /etc/group`" ]
then
  echo "Creating group $GROUP:"
  addgroup $GROUP
else
```

```
  echo "WARN: Group $GROUP already exists. Will not create it"
fi
# Same for the user
if [ -z "‘grep $USER /etc/passwd‘" ]
then
  echo "Creating user $USER:"
  adduser --system --home /home/$USER \
    --no-create-home --ingroup $GROUP \
    --disabled-password --disabled-login $USER
else
  echo "WARN: The user $USER already exists. Will not create it"
fi

# Change the init.d script

# First make a backup (check that there is not already
# one there first)
if [ ! -f $INITDBAK ]
then
  cp $INITD $INITDBAK
fi

# Then use it to change it
cat $INITDBAK |
eval $AWKS > $INITD

echo "WARN: The script $INITD has been changed, trying to test the changes.
echo "Restarting the named daemon (check for errors here)."

$INITD restart
if [ $? -ne 0 ]
then
  echo "ERR:  Failed to restart the daemon."
  restore
  exit 1
fi

RUNNING=‘ps eo fname |grep named‘
if [ -z "$RUNNING" ]
then
  echo "ERR:  Named is not running, probably due to a problem with the chan
  restore
  exit 1
fi

# Check if it’s running as expected
```

```
RUNUSER=‘ps eo user,fname |grep named |cut -f 1 -d " "‘

if [ "$RUNUSER" = "$USER" ]
then
  echo "All has gone well, named seems to be running now as $USER."
else
  echo "ERR:  The script failed to automatically change the system."
  echo "ERR:  Named is currently running as $RUNUSER."
  restore
  exit 1
fi

exit 0
```

Dieses Skript, wenn es auf Woodys (Debian 3.0) angepassten `Bind` angewendet wird, wird die folgende initd-Datei erstellen, nachdem der Nutzer und die Gruppe “named” erstellt wurde:

```
#!/bin/sh

PATH=/sbin:/bin:/usr/sbin:/usr/bin

test -x /usr/sbin/named || exit 0

start () {
  echo -n "Starting domain name service: named"
  start-stop-daemon --start --quiet \
    --pidfile /var/run/named.pid --exec /usr/sbin/named
  echo "."
}

stop () {
  echo -n "Stopping domain name service: named"
  # --exec doesn’t catch daemons running deleted instances of named,
  # as in an upgrade.  Fortunately, --pidfile is only going to hit
  # things from the pidfile.
  start-stop-daemon --stop --quiet  \
    --pidfile /var/run/named.pid --name named
  echo "."
}

case "$1" in
    start)
      start
    ;;
```

```
    stop)
      stop
    ;;

    restart|force-reload)
      stop
      sleep 2
      start
    ;;

    reload)
      /usr/sbin/ndc reload
    ;;

    *)
      echo "Usage: /etc/init.d/bind {start|stop|reload|restart|force-reload
      exit 1
    ;;
esac

exit 0
```

Anhang F

Schutz der Sicherheitsaktualisierung durch eine Firewall

Nach einer Standard-Installation könnten immer noch Sicherheitslücken auf dem System vorhanden sein. Falls Sie die Aktualisierungen für die verwundbaren Paket nicht auf einem anderen System herunterladen können (oder security.debian.org zu lokalen Zwecken spiegeln können), müssen Sie sich mit dem Internet verbinden, um die Pakete herunterzuladen.

Wenn Sie sich jedoch mit dem Internet verbinden, setzen Sie Ihr System einer Gefahr aus. Wenn einer Ihrer lokalen Dienste angreifbar ist, könnten Sie kompromittiert sein, noch bevor die Aktualisierung beendet ist! Sie mögen dies paranoid finden, aber eine Analyse vom Honeynet Project (`http://www.honeynet.org`) zeigt tatsächlich, dass ein System in weniger als drei Tagen kompromittiert werden kann, sogar wenn das System gar nicht der Öffentlichkeit bekannt ist (d.h. nicht in DNS-Einträgen auftaucht).

Wenn Sie eine Aktualisierung Ihres Systems durchführen, das nicht von einem externen System (z.B. einer Firewall) geschützt ist, können Sie trotzdem eine lokale Firewall so konfigurieren, dass Sie nur die Sicherheitsaktualisierung selbst erlaubt. Das Beispiel unten zeigt, wie die lokalen Firewall-Fähigkeiten aufgesetzt werden müssen, um ein eingeschränktes Setup zu erreichen, in dem nur Verbindungen zu security.debian.org erlaubt werden, während der Rest geloggt wird.

FIXME: add IP address for security.debian.org (since otherwise you need DNS up to work) on /etc/hosts.

FIXME: test this setup to see if it works properly

FIXME: this will only work with HTTP URLs since ftp might need the ip_conntrack_ftp module, or use passive mode.

```
# iptables -F
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
```

```
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
# iptables -P INPUT DROP
# iptables -P FORWARD DROP
# iptables -P OUTPUT DROP
# iptables -A OUTPUT -d security.debian.org --dport 80 -j ACCEPT
# iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# iptables -A INPUT -p icmp -j ACCEPT
# iptables -A INPUT -j LOG
# iptables -A OUTPUT -j LOG
# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0          state RELATED,E
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
LOG        all  --  anywhere             anywhere           LOG level warni

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     80   --  anywhere             security.debian.org
LOG        all  --  anywhere             anywhere           LOG level warni
```

Anhang G

Chroot-Umgebung für SSH

Es ist eine schwere Aufgabe, eine eingeschränkte Umgebung für SSH zu erstellen. Das liegt zum einen an seinen Abhängigkeiten und zum anderen daran, dass SSH im Gegensatz zu anderen Servern den Benutzern eine entfernte Shell zur Verfügung stellt. Daher müssen Sie sich überlegen, welche Programme Benutzer in der Umgebung verwenden sollen.

Sie haben zwei Möglichkeiten, eine beschränkte entfernte Shell einzurichten:

- die SSH-Benutzer in ein Chroot-Gefängnis einsperren: Dazu müssen Sie den SSH-Daemon so konfigurieren, dass er Benutzer nach der Authentifizierung in ein Chroot-Gefängnis einsperrt, bevor sie eine Shell bekommen. Jeder Benutzer kann seine eigene Umgebung haben.
- den SSH-Server in ein Chroot-Gefängnis einsperren: Wenn die SSH-Anwendung sich selbst in einer Chroot-Umgebung befindet, sind auch alle Benutzer in diese Umgebung eingesperrt.

Die erste Möglichkeit hat den Vorteil, dass es möglich ist, sowohl unbeschränkte als auch beschränkte Benutzer zu haben. Falls Sie keine Setuid-Anwendungen in der Chroot-Umgebung zur Verfügung stellen, wird es schwieriger, aus dem Gefängnis auszubrechen. Allerdings müssen Sie gegebenenfalls Chroot-Umgebungen für jeden Benutzer einzeln einrichten. Außerdem ist die Konfiguration schwieriger, da es Zusammenarbeit mit dem SSH-Server erfordert. Die zweite Möglichkeit ist leichter zu verwirklichen und schützt vor dem Ausnutzen eines Exploits des SSH-Servers, da auch dieser im Chroot-Gefängnis ist. Jedoch müssen alle Benutzer die gleiche Chroot-Umgebung verwenden. Verschiedene Umgebungen für verschiedene Benutzer sind nicht möglich.

G.1 SSH-Benutzer in ein Chroot-Gefängnis einsperren

Sie können den SSH-Server so einrichten, dass er bestimmte Benutzer in eine Chroot-Umgebung einsperrt, so dass sie eine Shell mit nur einer beschränkten Anzahl von Anwendungen zur Verfügung haben.

G.1.1 Einsatz von libpam-chroot

Der wahrscheinlich leichteste Weg ist, das Paket libpam-chroot, das in Debian vorhanden ist, zu verwenden. Wenn Sie es installiert haben, müssen Sie:

- /etc/pam.d/ssh verändern, um dieses PAM-Modul zu verwenden. Fügen Sie dazu als letzte Zeile Folgendes ein[1]:

```
session    required   pam_chroot.so
```

- eine passende Chroot-Umgebung für die Benutzer einrichten. Sie können versuchen, die Skripte unter /usr/share/doc/libpam-chroot/examples/ zu verwenden, das Programm makejail benutzen[2] oder eine minimale Debian-Umgebung mit debootstrap aufsetzen. Stellen Sie sicher, dass die Umgebung die notwendigen Geräte enthält.[3]

- /etc/security/chroot.conf bearbeiten, damit die ausgewählten Nutzer in das Verzeichnis eingesperrt werden, das Sie zuvor eingerichtet haben. Sie sollten unabhängige Verzeichnisse für verschiedene Nutzer haben, damit sie weder das ganze System noch sich gegenseitig sehen können.

- SSH konfigurieren: Je nach der eingesetzten OpenSSH-Version funktioniert die Chroot-Umgebung sofort. Seit 3.6.1p2 wird die Funktion *do_pam_session()* aufgerufen, nachdem sshd seine Rechte abgelegt hat. Da chroot() Root-Rechte benötigt, wird es mit Rechtetrennung nicht funktionieren. Allerdings wurde in neueren OpenSSH-Versionen der PAM-Code verändert, so dass do_pam_session vor dem Ablegen der Rechte aufgerufen wird. Daher funktioniert es auch mit aktivierter Rechtetrennung. Falls Sie sie abschalten müssen, müssen Sie /etc/ssh/sshd_config so verändern:

```
UsePrivilegeSeparation no
```

[1] Sie können die Option *debug* verwenden. Damit wird der Fortschritt des Moduls unter *authpriv.notice* protokolliert.

[2] Mit folgendem Python-Aufruf können Sie eine sehr eingeschränkte Bash-Umgebung für makejail erstellen. Erstellen Sie das Verzeichnis /var/chroots/users/foo und eine Datei mit dem Namen bash.py und folgendem Inhalt:

```
chroot="/var/chroots/user/foo"cleanJailFirst=1 testCommandsInsideJail=["bash ls"]
```

Führen Sie dann *makejail bash.py* aus, um eine Benutzer-Umgebung unter /var/chroots/user/foo zu erstellen. So testen Sie die Umgebung:

```
# chroot /var/chroots/user/foo/ ls bin dev etc lib proc sbin usr
```

[3] Unter Umständen benötigen Sie die Geräte /dev/ptmx und /dev/pty* und das Unterverzeichnis /dev/pts/. Es sollte ausreichen, MAKEDEV im /dev-Verzeichnis der Chroot-Umgebung auszuführen, um sie zu erstellen, falls sie nicht existieren. Wenn Sie einen Kernel einsetzen, der die Gerätedateien dynamisch erstellt (Version 2.6), müssen Sie die Dateien /dev/pts/ selbst erstellen und mit den passenden Rechten ausstatten.

> Beachten Sie, dass das die Sicherheit Ihres Systems verringern wird, da dann der OpenSSH-Server als *Root* laufen wird. Das bedeutet, dass wenn eine Angriffsmöglichkeit aus der Ferne gegen OpenSSH entdeckt wird, ein Angreifer *Root*-Rechte anstatt nur *Sshd*-Rechte erlangen wird und somit das gesamte System kompromittiert.[4]

Wenn Sie die *Rechtetrennung* nicht deaktivieren, brauchen Sie im Chroot-Gefängnis `/etc/passwd`, welches die Benutzer-UID enthält, damit die *Rechtetrennung* funktioniert.

Wenn Sie die Option *Rechtetrennung* auf **yes** gesetzt haben und Ihre Version von OpenSSH nicht richtig läuft, müssen Sie sie abschalten. Wenn Sie das unterlassen, werden Benutzer, die sich mit Ihrem Server verbinden wollen und von diesem Modul in eine Chroot-Umgebung eingesperrt werden sollen, Folgendes zu sehen bekommen:

```
$ ssh -l user server
user@server's password:
Connection to server closed by remote host.
Connection to server closed.
```

Dies geschieht, weil der SSH-Daemon, der als 'sshd' läuft, nicht den Systemaufruf chroot() ausführen kann. Um die Rechtetrennung abzuschalten, müssen Sie die Konfigurationsdatei `/etc/ssh/sshd_config` wie oben beschrieben verändern.

Beachten Sie, dass, wenn Folgendes fehlt, sich die Benutzer nicht in der Chroot-Umgebung anmelden können:

- Das Dateisystem `/proc` muss in der Chroot-Umgebung des Benutzers gemountet sein.
- Die notwendigen Geräte unter `/dev/pts/` müssen vorliegen. Falls diese Dateien automatisch vom Kernel erstellt werden, müssen Sie sie von Hand unter `/dev/` in der Chroot-Umgebung erstellen.
- Das Home-Verzeichnis des Benutzers muss in der Chroot-Umgebung existieren. Ansonsten wird der SSH-Daemon nicht fortfahren.

Sie können diese Probleme mit dem Schlüsselwort *debug* in der PAM-Konfiguration `/etc/pam.d/ssh` debuggen. Falls Sie auf Probleme stoßen, kann es sich als nützlich erweisen, auch den Debugging-Modus des SSH-Clients zu aktivieren.

Hinweis: Diese Informationen sind auch in `/usr/share/doc/libpam-chroot/README.Debian.gz` enthalten (und vielleicht aktueller). Bitte überprüfen Sie, ob dort aktualisierte Informationen vorhanden sind, bevor Sie die oben aufgezeigten Schritte ausführen.

[4] Wenn Sie einen Kernel verwenden, der Mandatory-Access-Control (RSBAC/SElinux) unterstützt, müssen Sie die Konfiguration nicht ändern, wenn Sie dem *Sshd*-Benutzer die notwendigen Rechte einräumen, um den Systemaufruf chroot() ausführen zu können.

G.1.2 Patchen des ssh-Servers

Debians sshd gestattet nicht, die Bewegungen eines Benutzer durch den Server zu beschränken, da er keine Chroot-Funktionalität besitzt. Diese ist im Gegensatz dazu Bestandteil des kommerziellen Programms sshd2 (es verwendet 'ChrootGroups' oder 'ChrootUsers', siehe sshd2_config(5)). Allerdings gibt es einen Patch, der sshd um diese Funktion erweitert. Den Patch erhalten Sie unter ChrootSSH project (http://chrootssh.sourceforge.net) (wurde in Fehlerbericht #139047 (http://bugs.debian.org/139047) nachgefragt). Der Patch könnte Bestandteil von zukünftigen Veröffentlichungen des OpenSSH-Pakets werden. Emmanuel Lacour bietet ssh-Pakete als Debs mit diesen Fähigkeiten für *Sarge* an. Sie sind unter http://debian.home-dn.net/sarge/ssh/ verfügbar. Beachten Sie aber, dass sie nicht aktuell sein müssen, daher wird empfohlen, den Weg der Kompilierung zu gehen.

Eine Beschreibung der notwendigen Schritte finden Sie unter http://mail.incredimail.com/howto/openssh/ (so ziemlich alles dort trifft auch auf Debian zu, auch wenn von Red Hat 7.3 die Rede ist). Nachdem Sie den Patch angewendet haben, müssen Sie /etc/passwd anpassen und darin das Home-Verzeichnis der Benutzer ändern (mit dem speziellen /./ Kürzel).

```
joeuser:x:1099:1099:Joe Random User:/home/joe/./:/bin/bash
```

Dies wird *sowohl* den Fernzugriff auf die Shell *als auch* Fernkopien über den ssh-Kanal einschränken.

Gehen Sie sicher, dass Sie alle benötigten Programme und Bibliotheken in den Chroot-Pfaden der Benutzer haben. Diese Dateien sollten Root als Eigentümer haben, um Manipulationen durch den Benutzer zu verhindern (zum Beispiel um das chroot-Gefängnis zu verlassen). Ein Beispiel könnte so aussehen:

```
./bin:
total 660
drwxr-xr-x    2 root     root         4096 Mar 18 13:36 .
drwxr-xr-x    8 guest    guest        4096 Mar 15 16:53 ..
-r-xr-xr-x    1 root     root       531160 Feb  6 22:36 bash
-r-xr-xr-x    1 root     root        43916 Nov 29 13:19 ls
-r-xr-xr-x    1 root     root        16684 Nov 29 13:19 mkdir
-rwxr-xr-x    1 root     root        23960 Mar 18 13:36 more
-r-xr-xr-x    1 root     root         9916 Jul 26  2001 pwd
-r-xr-xr-x    1 root     root        24780 Nov 29 13:19 rm
lrwxrwxrwx    1 root     root            4 Mar 30 16:29 sh -> bash

./etc:
total 24
drwxr-xr-x    2 root     root         4096 Mar 15 16:13 .
drwxr-xr-x    8 guest    guest        4096 Mar 15 16:53 ..
-rw-r--r--    1 root     root           54 Mar 15 13:23 group
```

```
-rw-r--r--    1 root     root           428 Mar 15 15:56 hosts
-rw-r--r--    1 root     root            44 Mar 15 15:53 passwd
-rw-r--r--    1 root     root            52 Mar 15 13:23 shells

./lib:
total 1848
drwxr-xr-x    2 root     root          4096 Mar 18 13:37 .
drwxr-xr-x    8 guest    guest         4096 Mar 15 16:53 ..
-rwxr-xr-x    1 root     root         92511 Mar 15 12:49 ld-linux.so.2
-rwxr-xr-x    1 root     root       1170812 Mar 15 12:49 libc.so.6
-rw-r--r--    1 root     root         20900 Mar 15 13:01 libcrypt.so.1
-rw-r--r--    1 root     root          9436 Mar 15 12:49 libdl.so.2
-rw-r--r--    1 root     root        248132 Mar 15 12:48 libncurses.so.5
-rw-r--r--    1 root     root         71332 Mar 15 13:00 libnsl.so.1
-rw-r--r--    1 root     root         34144 Mar 15 16:10
libnss_files.so.2
-rw-r--r--    1 root     root         29420 Mar 15 12:57 libpam.so.0
-rw-r--r--    1 root     root        105498 Mar 15 12:51 libpthread.so.0
-rw-r--r--    1 root     root         25596 Mar 15 12:51 librt.so.1
-rw-r--r--    1 root     root          7760 Mar 15 12:59 libutil.so.1
-rw-r--r--    1 root     root         24328 Mar 15 12:57 libwrap.so.0

./usr:
total 16
drwxr-xr-x    4 root     root          4096 Mar 15 13:00 .
drwxr-xr-x    8 guest    guest         4096 Mar 15 16:53 ..
drwxr-xr-x    2 root     root          4096 Mar 15 15:55 bin
drwxr-xr-x    2 root     root          4096 Mar 15 15:37 lib

./usr/bin:
total 340
drwxr-xr-x    2 root     root          4096 Mar 15 15:55 .
drwxr-xr-x    4 root     root          4096 Mar 15 13:00 ..
-rwxr-xr-x    1 root     root         10332 Mar 15 15:55 env
-rwxr-xr-x    1 root     root         13052 Mar 15 13:13 id
-r-xr-xr-x    1 root     root         25432 Mar 15 12:40 scp
-rwxr-xr-x    1 root     root         43768 Mar 15 15:15 sftp
-r-sr-xr-x    1 root     root        218456 Mar 15 12:40 ssh
-rwxr-xr-x    1 root     root          9692 Mar 15 13:17 tty

./usr/lib:
total 852
drwxr-xr-x    2 root     root          4096 Mar 15 15:37 .
drwxr-xr-x    4 root     root          4096 Mar 15 13:00 ..
-rw-r--r--    1 root     root        771088 Mar 15 13:01
```

```
libcrypto.so.0.9.6
-rw-r--r--    1 root     root        54548 Mar 15 13:00 libz.so.1
-rwxr-xr-x    1 root     root        23096 Mar 15 15:37 sftp-server
```

G.2 Einsperren des SSH-Servers in einem Chroot-Gefängnis

Wenn Sie eine Chroot-Umgebung erstellen, welche die Dateien des SSH-Servers enthält, z.B. unter `/var/chroot/ssh`, sollten Sie den im `chroot`-Gefängnis eingesperrten `ssh`-Server mit diesem Befehl starten:

```
# chroot /var/chroot/ssh /sbin/sshd -f /etc/sshd_config
```

Das führt dazu, dass der `sshd`-Daemon innerhalb des Chroot-Gefängnisses gestartet wird. Dazu müssen Sie zunächst dafür sorgen, dass das Verzeichnis `/var/chroot/ssh` den SSH-Server und die Werkzeuge enthält, die Benutzer benötigen, die mit dem Server verbunden sind. Wenn Sie das vorhaben, sollten Sie sicherstellen, dass OpenSSH Rechtetrennung einsetzt (was standardmäßig so ist). Dazu muss in der Konfigurationsdatei `/etc/ssh/sshd_config` folgende Zeile enthalten sein:

```
UsePrivilegeSeparation yes
```

Dadurch wird der entfernte Daemon so wenig Dinge wie möglich als Root ausführen. Wenn er also einen Fehler enthalten sollte, kann damit nicht aus dem Chroot-Gefängnis ausgebrochen werden. Beachten Sie, dass, anders als wenn Sie eine Chroot-Umgebung für jeden Benutzer einzeln einrichten, in diesem Fall der SSH-Daemon im selben Chroot-Gefängnis wie die Benutzer läuft. Es gibt also mindestens einen Prozess in der Chroot-Umgebung, der als Root läuft. Mit ihm ist es möglich, aus dem Chroot-Gefängnis auszubrechen.

Beachten Sie auch, dass SSH nur funktioniert, wenn die Partition, auf der die Chroot-Umgebung eingerichtet wurde, nicht mit der Option *nodev* gemountet wurde. Wenn Sie diese Option verwenden, bekommen Sie folgende Fehlermeldung: *PRNG is not seeded*, weil `/dev/urandom` nicht in der Chroot-Umgebung funktioniert.

G.2.1 Einrichten eines minimalen Systems (der wirklich leichte Weg)

Sie können mit `debootstrap` eine minimale Umgebung einrichten, die ausschließlich den SSH-Server enthält. Dafür müssen Sie nur eine Chroot-Umgebung einrichten, wie es im Chroot-Abschnitt der Debian-Referenz (`http://www.de.debian.org/doc/manuals/reference/ch-tips#s-chroot`) beschrieben wird. Diese Vorgehensweise ist idiotensicher (Sie werden alle für die Chroot-Umgebung notwendigen Bestandteile erhalten), aber dies geht auf Kosten von Plattenspeicher. Eine minimale Installation von Debian benötigt einige hundert Megabyte. Dieses minimale System könnte auch Setuid-Dateien enthalten, mit denen ein Benutzer aus dem Chroot-Gefängnis ausbrechen könnte, wenn sie eine Rechteerweiterung zulassen.

G.2.2 Automatisches Erstellen der Umgebung (der leichte Weg)

Mit dem Paket `makejail` können Sie leicht eine eingeschränkte Umgebung erstellen, da es automatisch den Trace des Server-Daemons verfolgt (mit `strace`) und dafür sorgt, dass er in der eingeschränkten Umgebung läuft.

Der Vorteil von Programmen, die automatisch die `chroot`-Umgebung einrichten, liegt darin, dass sie im Stande sind, Pakete in die `chroot`-Umgebung zu kopieren (und verfolgen sogar die Abhängigkeiten der Pakete, um sicherzustellen, dass sie vollständig sind). Dadurch wird das Bereitstellen von Anwendungen für Benutzer leichter.

Um ein Chroot-Gefängnis aus den von `makejail` zur Verfügung gestellten Beispielen einzurichten, müssen Sie `/var/chroot/sshd` erstellen und folgenden Befehl ausführen:

```
# makejail /usr/share/doc/makejail/examples/sshd.py
```

Dies wird eine Chroot-Umgebung im Verzeichnis `/var/chroot/sshd` erstellen. Beachten Sie, dass diese Chroot-Umgebung nicht voll funktionstüchtig ist, bis Sie:

- Das Dateisystem *procfs* in `/var/chroot/sshd/proc` eingebunden haben. `Makejail` wird es für Sie einbinden. Aber nach einem Neustart werden Sie es erneut einbinden müssen:

  ```
  # mount -t proc proc /var/chroot/sshd/proc
  ```

  Es kann auch automatisch eingebunden werden. Dazu müssen Sie `/etc/fstab` bearbeiten und folgende Zeile eintragen:

  ```
  proc-ssh /var/chroot/sshd/proc  proc none 0 0
  ```

- Syslog auf das Geräte `/dev/log` in der Chroot-Umgebung horchen lassen. Dazu müssen Sie `/etc/default/syslogd` ändern und *-a /var/chroot/sshd/dev/log* zur Definition der Variablen **SYSLOGD** hinzufügen.

Sehen Sie sich die Beispielsdatei an, um herauszufinden, welche Änderungen an der Umgebung vorgenommen werden müssen. Einige diese Änderungen können nicht automatisch vorgenommen werden, wie z.B. das Kopieren des Home-Verzeichnisses eines Benutzers. Außerdem sollten Sie die Gefährdung von sensiblen Informationen begrenzen, indem Sie nur die Daten bestimmter Benutzer aus den Dateien `/etc/shadow` und `/etc/group` kopieren. Beachten Sie, dass, falls Sie Rechtetrennung verwenden, der Benutzer *sshd* in diesen Dateien vorhanden sein muss.

Die folgende Beispielumgebung wurde (ein wenig) unter Debian 3.0 getestet. Sie basiert auf der Konfigurationsdatei, die mit dem Paket geliefert wird, und beinhaltet das Paket `fileutils`.

```
.
|-- bin
|   |-- ash
|   |-- bash
|   |-- chgrp
|   |-- chmod
|   |-- chown
|   |-- cp
|   |-- csh -> /etc/alternatives/csh
|   |-- dd
|   |-- df
|   |-- dir
|   |-- fdflush
|   |-- ksh
|   |-- ln
|   |-- ls
|   |-- mkdir
|   |-- mknod
|   |-- mv
|   |-- rbash -> bash
|   |-- rm
|   |-- rmdir
|   |-- sh -> bash
|   |-- sync
|   |-- tcsh
|   |-- touch
|   |-- vdir
|   |-- zsh -> /etc/alternatives/zsh
|   `-- zsh4
|-- dev
|   |-- null
|   |-- ptmx
|   |-- pts
|   |-- ptya0
(...)
|   |-- tty
|   |-- tty0
(...)
|   `-- urandom
|-- etc
|   |-- alternatives
|   |   |-- csh -> /bin/tcsh
|   |   `-- zsh -> /bin/zsh4
|   |-- environment
|   |-- hosts
|   |-- hosts.allow
```

```
|   |-- hosts.deny
|   |-- ld.so.conf
|   |-- localtime -> /usr/share/zoneinfo/Europe/Madrid
|   |-- motd
|   |-- nsswitch.conf
|   |-- pam.conf
|   |-- pam.d
|   |   |-- other
|   |   `-- ssh
|   |-- passwd
|   |-- resolv.conf
|   |-- security
|   |   |-- access.conf
|   |   |-- chroot.conf
|   |   |-- group.conf
|   |   |-- limits.conf
|   |   |-- pam_env.conf
|   |   `-- time.conf
|   |-- shadow
|   |-- shells
|   `-- ssh
|       |-- moduli
|       |-- ssh_host_dsa_key
|       |-- ssh_host_dsa_key.pub
|       |-- ssh_host_rsa_key
|       |-- ssh_host_rsa_key.pub
|       `-- sshd_config
|-- home
|   `-- userX
|-- lib
|   |-- ld-2.2.5.so
|   |-- ld-linux.so.2 -> ld-2.2.5.so
|   |-- libc-2.2.5.so
|   |-- libc.so.6 -> libc-2.2.5.so
|   |-- libcap.so.1 -> libcap.so.1.10
|   |-- libcap.so.1.10
|   |-- libcrypt-2.2.5.so
|   |-- libcrypt.so.1 -> libcrypt-2.2.5.so
|   |-- libdl-2.2.5.so
|   |-- libdl.so.2 -> libdl-2.2.5.so
|   |-- libm-2.2.5.so
|   |-- libm.so.6 -> libm-2.2.5.so
|   |-- libncurses.so.5 -> libncurses.so.5.2
|   |-- libncurses.so.5.2
|   |-- libnsl-2.2.5.so
|   |-- libnsl.so.1 -> libnsl-2.2.5.so
```

```
|   |-- libnss_compat-2.2.5.so
|   |-- libnss_compat.so.2 -> libnss_compat-2.2.5.so
|   |-- libnss_db-2.2.so
|   |-- libnss_db.so.2 -> libnss_db-2.2.so
|   |-- libnss_dns-2.2.5.so
|   |-- libnss_dns.so.2 -> libnss_dns-2.2.5.so
|   |-- libnss_files-2.2.5.so
|   |-- libnss_files.so.2 -> libnss_files-2.2.5.so
|   |-- libnss_hesiod-2.2.5.so
|   |-- libnss_hesiod.so.2 -> libnss_hesiod-2.2.5.so
|   |-- libnss_nis-2.2.5.so
|   |-- libnss_nis.so.2 -> libnss_nis-2.2.5.so
|   |-- libnss_nisplus-2.2.5.so
|   |-- libnss_nisplus.so.2 -> libnss_nisplus-2.2.5.so
|   |-- libpam.so.0 -> libpam.so.0.72
|   |-- libpam.so.0.72
|   |-- libpthread-0.9.so
|   |-- libpthread.so.0 -> libpthread-0.9.so
|   |-- libresolv-2.2.5.so
|   |-- libresolv.so.2 -> libresolv-2.2.5.so
|   |-- librt-2.2.5.so
|   |-- librt.so.1 -> librt-2.2.5.so
|   |-- libutil-2.2.5.so
|   |-- libutil.so.1 -> libutil-2.2.5.so
|   |-- libwrap.so.0 -> libwrap.so.0.7.6
|   |-- libwrap.so.0.7.6
|   `-- security
|       |-- pam_access.so
|       |-- pam_chroot.so
|       |-- pam_deny.so
|       |-- pam_env.so
|       |-- pam_filter.so
|       |-- pam_ftp.so
|       |-- pam_group.so
|       |-- pam_issue.so
|       |-- pam_lastlog.so
|       |-- pam_limits.so
|       |-- pam_listfile.so
|       |-- pam_mail.so
|       |-- pam_mkhomedir.so
|       |-- pam_motd.so
|       |-- pam_nologin.so
|       |-- pam_permit.so
|       |-- pam_rhosts_auth.so
|       |-- pam_rootok.so
|       |-- pam_securetty.so
```

```
|       |-- pam_shells.so
|       |-- pam_stress.so
|       |-- pam_tally.so
|       |-- pam_time.so
|       |-- pam_unix.so
|       |-- pam_unix_acct.so -> pam_unix.so
|       |-- pam_unix_auth.so -> pam_unix.so
|       |-- pam_unix_passwd.so -> pam_unix.so
|       |-- pam_unix_session.so -> pam_unix.so
|       |-- pam_userdb.so
|       |-- pam_warn.so
|       `-- pam_wheel.so
|-- sbin
|   `-- start-stop-daemon
|-- usr
|   |-- bin
|   |   |-- dircolors
|   |   |-- du
|   |   |-- install
|   |   |-- link
|   |   |-- mkfifo
|   |   |-- shred
|   |   |-- touch -> /bin/touch
|   |   `-- unlink
|   |-- lib
|   |   |-- libcrypto.so.0.9.6
|   |   |-- libdb3.so.3 -> libdb3.so.3.0.2
|   |   |-- libdb3.so.3.0.2
|   |   |-- libz.so.1 -> libz.so.1.1.4
|   |   `-- libz.so.1.1.4
|   |-- sbin
|   |   `-- sshd
|   `-- share
|       |-- locale
|       |   `-- es
|       |       |-- LC_MESSAGES
|       |       |   |-- fileutils.mo
|       |       |   |-- libc.mo
|       |       |   `-- sh-utils.mo
|       |       `-- LC_TIME -> LC_MESSAGES
|       `-- zoneinfo
|           `-- Europe
|               `-- Madrid
`-- var
    `-- run
        |-- sshd
```

```
        '-- sshd.pid

27 directories, 733 files
```

Bei Debian 3.1 müssen Sie sicherstellen, dass das Gefängnis auch die Dateien für PAM enthält. Falls es nicht schon `makejail` für Sie erledigt hat, müssen Sie folgende Dateien in die Chroot-Umgebung kopiert:

```
$ ls /etc/pam.d/common-*
/etc/pam.d/common-account  /etc/pam.d/common-password
/etc/pam.d/common-auth     /etc/pam.d/common-session
```

G.2.3 Die Chroot-Umgebung von Hand erstellen (der schwierige Weg)

Es ist möglich, eine Umgebung mit der Trial-and-Error-Methode zu erstellen. Dazu müssen Sie die Traces und die Logdateien des `sshd`-Servers überwachen, um die notwendigen Dateien herauszufinden. Die folgende Umgebung, die von José Luis Ledesma zur Verfügung gestellt wurde, ist eine beispielhafte Auflistung der Dateien in einer `chroot`-Umgebung für `ssh` unter Debian 3.0:[5]

```
.:
total 36
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ./
drwxr-xr-x 11 root root 4096 Jun 3 13:43 ../
drwxr-xr-x 2 root root 4096 Jun 4 12:13 bin/
drwxr-xr-x 2 root root 4096 Jun 4 12:16 dev/
drwxr-xr-x 4 root root 4096 Jun 4 12:35 etc/
drwxr-xr-x 3 root root 4096 Jun 4 12:13 lib/
drwxr-xr-x 2 root root 4096 Jun 4 12:35 sbin/
drwxr-xr-x 2 root root 4096 Jun 4 12:32 tmp/
drwxr-xr-x 2 root root 4096 Jun 4 12:16 usr/
./bin:
total 8368
drwxr-xr-x 2 root root 4096 Jun 4 12:13 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rwxr-xr-x 1 root root 109855 Jun 3 13:45 a2p*
-rwxr-xr-x 1 root root 387764 Jun 3 13:45 bash*
-rwxr-xr-x 1 root root 36365 Jun 3 13:45 c2ph*
-rwxr-xr-x 1 root root 20629 Jun 3 13:45 dprofpp*
-rwxr-xr-x 1 root root 6956 Jun 3 13:46 env*
-rwxr-xr-x 1 root root 158116 Jun 3 13:45 fax2ps*
```

[5] Beachten Sie, dass keine SETUID-Dateien vorhanden sind. Das erschwert es entfernten Benutzern, aus der `chroot`-Umgebung auszubrechen. Es verhindert allerdings auch, dass Nutzer ihr Passwort ändern, da `passwd` nicht die Dateien `/etc/passwd` und `/etc/shadow` verändern kann.

```
-rwxr-xr-x 1 root root 104008 Jun 3 13:45 faxalter*
-rwxr-xr-x 1 root root 89340 Jun 3 13:45 faxcover*
-rwxr-xr-x 1 root root 441584 Jun 3 13:45 faxmail*
-rwxr-xr-x 1 root root 96036 Jun 3 13:45 faxrm*
-rwxr-xr-x 1 root root 107000 Jun 3 13:45 faxstat*
-rwxr-xr-x 1 root root 77832 Jun 4 11:46 grep*
-rwxr-xr-x 1 root root 19597 Jun 3 13:45 h2ph*
-rwxr-xr-x 1 root root 46979 Jun 3 13:45 h2xs*
-rwxr-xr-x 1 root root 10420 Jun 3 13:46 id*
-rwxr-xr-x 1 root root 4528 Jun 3 13:46 ldd*
-rwxr-xr-x 1 root root 111386 Jun 4 11:46 less*
-r-xr-xr-x 1 root root 26168 Jun 3 13:45 login*
-rwxr-xr-x 1 root root 49164 Jun 3 13:45 ls*
-rwxr-xr-x 1 root root 11600 Jun 3 13:45 mkdir*
-rwxr-xr-x 1 root root 24780 Jun 3 13:45 more*
-rwxr-xr-x 1 root root 154980 Jun 3 13:45 pal2rgb*
-rwxr-xr-x 1 root root 27920 Jun 3 13:46 passwd*
-rwxr-xr-x 1 root root 4241 Jun 3 13:45 pl2pm*
-rwxr-xr-x 1 root root 2350 Jun 3 13:45 pod2html*
-rwxr-xr-x 1 root root 7875 Jun 3 13:45 pod2latex*
-rwxr-xr-x 1 root root 17587 Jun 3 13:45 pod2man*
-rwxr-xr-x 1 root root 6877 Jun 3 13:45 pod2text*
-rwxr-xr-x 1 root root 3300 Jun 3 13:45 pod2usage*
-rwxr-xr-x 1 root root 3341 Jun 3 13:45 podchecker*
-rwxr-xr-x 1 root root 2483 Jun 3 13:45 podselect*
-r-xr-xr-x 1 root root 82412 Jun 4 11:46 ps*
-rwxr-xr-x 1 root root 36365 Jun 3 13:45 pstruct*
-rwxr-xr-x 1 root root 7120 Jun 3 13:45 pwd*
-rwxr-xr-x 1 root root 179884 Jun 3 13:45 rgb2ycbcr*
-rwxr-xr-x 1 root root 20532 Jun 3 13:45 rm*
-rwxr-xr-x 1 root root 6720 Jun 4 10:15 rmdir*
-rwxr-xr-x 1 root root 14705 Jun 3 13:45 s2p*
-rwxr-xr-x 1 root root 28764 Jun 3 13:46 scp*
-rwxr-xr-x 1 root root 385000 Jun 3 13:45 sendfax*
-rwxr-xr-x 1 root root 67548 Jun 3 13:45 sendpage*
-rwxr-xr-x 1 root root 88632 Jun 3 13:46 sftp*
-rwxr-xr-x 1 root root 387764 Jun 3 13:45 sh*
-rws--x--x 1 root root 744500 Jun 3 13:46 slogin*
-rwxr-xr-x 1 root root 14523 Jun 3 13:46 splain*
-rws--x--x 1 root root 744500 Jun 3 13:46 ssh*
-rwxr-xr-x 1 root root 570960 Jun 3 13:46 ssh-add*
-rwxr-xr-x 1 root root 502952 Jun 3 13:46 ssh-agent*
-rwxr-xr-x 1 root root 575740 Jun 3 13:46 ssh-keygen*
-rwxr-xr-x 1 root root 383480 Jun 3 13:46 ssh-keyscan*
-rwxr-xr-x 1 root root 39 Jun 3 13:46 ssh_europa*
-rwxr-xr-x 1 root root 107252 Jun 4 10:14 strace*
```

```
-rwxr-xr-x 1 root root 8323 Jun 4 10:14 strace-graph*
-rwxr-xr-x 1 root root 158088 Jun 3 13:46 thumbnail*
-rwxr-xr-x 1 root root 6312 Jun 3 13:46 tty*
-rwxr-xr-x 1 root root 55904 Jun 4 11:46 useradd*
-rwxr-xr-x 1 root root 585656 Jun 4 11:47 vi*
-rwxr-xr-x 1 root root 6444 Jun 4 11:45 whoami*
./dev:
total 8
drwxr-xr-x 2 root root 4096 Jun 4 12:16 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
crw-r--r-- 1 root root 1, 9 Jun 3 13:43 urandom
./etc:
total 208
drwxr-xr-x 4 root root 4096 Jun 4 12:35 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rw------- 1 root root 0 Jun 4 11:46 .pwd.lock
-rw-r--r-- 1 root root 653 Jun 3 13:46 group
-rw-r--r-- 1 root root 242 Jun 4 11:33 host.conf
-rw-r--r-- 1 root root 857 Jun 4 12:04 hosts
-rw-r--r-- 1 root root 1050 Jun 4 11:29 ld.so.cache
-rw-r--r-- 1 root root 304 Jun 4 11:28 ld.so.conf
-rw-r--r-- 1 root root 235 Jun 4 11:27 ld.so.conf~
-rw-r--r-- 1 root root 88039 Jun 3 13:46 moduli
-rw-r--r-- 1 root root 1342 Jun 4 11:34 nsswitch.conf
drwxr-xr-x 2 root root 4096 Jun 4 12:02 pam.d/
-rw-r--r-- 1 root root 28 Jun 4 12:00 pam_smb.conf
-rw-r--r-- 1 root root 2520 Jun 4 11:57 passwd
-rw-r--r-- 1 root root 7228 Jun 3 13:48 profile
-rw-r--r-- 1 root root 1339 Jun 4 11:33 protocols
-rw-r--r-- 1 root root 274 Jun 4 11:44 resolv.conf
drwxr-xr-x 2 root root 4096 Jun 3 13:43 security/
-rw-r----- 1 root root 1178 Jun 4 11:51 shadow
-rw------- 1 root root 80 Jun 4 11:45 shadow-
-rw-r----- 1 root root 1178 Jun 4 11:48 shadow.old
-rw-r--r-- 1 root root 161 Jun 3 13:46 shells
-rw-r--r-- 1 root root 1144 Jun 3 13:46 ssh_config
-rw------- 1 root root 668 Jun 3 13:46 ssh_host_dsa_key
-rw-r--r-- 1 root root 602 Jun 3 13:46 ssh_host_dsa_key.pub
-rw------- 1 root root 527 Jun 3 13:46 ssh_host_key
-rw-r--r-- 1 root root 331 Jun 3 13:46 ssh_host_key.pub
-rw------- 1 root root 883 Jun 3 13:46 ssh_host_rsa_key
-rw-r--r-- 1 root root 222 Jun 3 13:46 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root 2471 Jun 4 12:15 sshd_config
./etc/pam.d:
total 24
drwxr-xr-x 2 root root 4096 Jun 4 12:02 ./
```

```
drwxr-xr-x 4 root root 4096 Jun 4 12:35 ../
lrwxrwxrwx 1 root root 4 Jun 4 12:02 other -> sshd
-rw-r--r-- 1 root root 318 Jun 3 13:46 passwd
-rw-r--r-- 1 root root 546 Jun 4 11:36 ssh
-rw-r--r-- 1 root root 479 Jun 4 12:02 sshd
-rw-r--r-- 1 root root 370 Jun 3 13:46 su
./etc/security:
total 32
drwxr-xr-x 2 root root 4096 Jun 3 13:43 ./
drwxr-xr-x 4 root root 4096 Jun 4 12:35 ../
-rw-r--r-- 1 root root 1971 Jun 3 13:46 access.conf
-rw-r--r-- 1 root root 184 Jun 3 13:46 chroot.conf
-rw-r--r-- 1 root root 2145 Jun 3 13:46 group.conf
-rw-r--r-- 1 root root 1356 Jun 3 13:46 limits.conf
-rw-r--r-- 1 root root 2858 Jun 3 13:46 pam_env.conf
-rw-r--r-- 1 root root 2154 Jun 3 13:46 time.conf
./lib:
total 8316
drwxr-xr-x 3 root root 4096 Jun 4 12:13 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rw-r--r-- 1 root root 1024 Jun 4 11:51 cracklib_dict.hwm
-rw-r--r-- 1 root root 214324 Jun 4 11:51 cracklib_dict.pwd
-rw-r--r-- 1 root root 11360 Jun 4 11:51 cracklib_dict.pwi
-rwxr-xr-x 1 root root 342427 Jun 3 13:46 ld-linux.so.2*
-rwxr-xr-x 1 root root 4061504 Jun 3 13:46 libc.so.6*
lrwxrwxrwx 1 root root 15 Jun 4 12:11 libcrack.so -> libcrack.so.2.7*
lrwxrwxrwx 1 root root 15 Jun 4 12:11 libcrack.so.2 -> libcrack.so.2.7*
-rwxr-xr-x 1 root root 33291 Jun 4 11:39 libcrack.so.2.7*
-rwxr-xr-x 1 root root 60988 Jun 3 13:46 libcrypt.so.1*
-rwxr-xr-x 1 root root 71846 Jun 3 13:46 libdl.so.2*
-rwxr-xr-x 1 root root 27762 Jun 3 13:46 libhistory.so.4.0*
lrwxrwxrwx 1 root root 17 Jun 4 12:12 libncurses.so.4 -> libncurses.so.4.2*
-rwxr-xr-x 1 root root 503903 Jun 3 13:46 libncurses.so.4.2*
lrwxrwxrwx 1 root root 17 Jun 4 12:12 libncurses.so.5 -> libncurses.so.5.0*
-rwxr-xr-x 1 root root 549429 Jun 3 13:46 libncurses.so.5.0*
-rwxr-xr-x 1 root root 369801 Jun 3 13:46 libnsl.so.1*
-rwxr-xr-x 1 root root 142563 Jun 4 11:49 libnss_compat.so.1*
-rwxr-xr-x 1 root root 215569 Jun 4 11:49 libnss_compat.so.2*
-rwxr-xr-x 1 root root 61648 Jun 4 11:34 libnss_dns.so.1*
-rwxr-xr-x 1 root root 63453 Jun 4 11:34 libnss_dns.so.2*
-rwxr-xr-x 1 root root 63782 Jun 4 11:34 libnss_dns6.so.2*
-rwxr-xr-x 1 root root 205715 Jun 3 13:46 libnss_files.so.1*
-rwxr-xr-x 1 root root 235932 Jun 3 13:49 libnss_files.so.2*
-rwxr-xr-x 1 root root 204383 Jun 4 11:33 libnss_nis.so.1*
-rwxr-xr-x 1 root root 254023 Jun 4 11:33 libnss_nis.so.2*
-rwxr-xr-x 1 root root 256465 Jun 4 11:33 libnss_nisplus.so.2*
```

```
lrwxrwxrwx 1 root root 14 Jun 4 12:12 libpam.so.0 -> libpam.so.0.72*
-rwxr-xr-x 1 root root 31449 Jun 3 13:46 libpam.so.0.72*
lrwxrwxrwx 1 root root 19 Jun 4 12:12 libpam_misc.so.0 ->
libpam_misc.so.0.72*
-rwxr-xr-x 1 root root 8125 Jun 3 13:46 libpam_misc.so.0.72*
lrwxrwxrwx 1 root root 15 Jun 4 12:12 libpamc.so.0 -> libpamc.so.0.72*
-rwxr-xr-x 1 root root 10499 Jun 3 13:46 libpamc.so.0.72*
-rwxr-xr-x 1 root root 176427 Jun 3 13:46 libreadline.so.4.0*
-rwxr-xr-x 1 root root 44729 Jun 3 13:46 libutil.so.1*
-rwxr-xr-x 1 root root 70254 Jun 3 13:46 libz.a*
lrwxrwxrwx 1 root root 13 Jun 4 12:13 libz.so -> libz.so.1.1.3*
lrwxrwxrwx 1 root root 13 Jun 4 12:13 libz.so.1 -> libz.so.1.1.3*
-rwxr-xr-x 1 root root 63312 Jun 3 13:46 libz.so.1.1.3*
drwxr-xr-x 2 root root 4096 Jun 4 12:00 security/
./lib/security:
total 668
drwxr-xr-x 2 root root 4096 Jun 4 12:00 ./
drwxr-xr-x 3 root root 4096 Jun 4 12:13 ../
-rwxr-xr-x 1 root root 10067 Jun 3 13:46 pam_access.so*
-rwxr-xr-x 1 root root 8300 Jun 3 13:46 pam_chroot.so*
-rwxr-xr-x 1 root root 14397 Jun 3 13:46 pam_cracklib.so*
-rwxr-xr-x 1 root root 5082 Jun 3 13:46 pam_deny.so*
-rwxr-xr-x 1 root root 13153 Jun 3 13:46 pam_env.so*
-rwxr-xr-x 1 root root 13371 Jun 3 13:46 pam_filter.so*
-rwxr-xr-x 1 root root 7957 Jun 3 13:46 pam_ftp.so*
-rwxr-xr-x 1 root root 12771 Jun 3 13:46 pam_group.so*
-rwxr-xr-x 1 root root 10174 Jun 3 13:46 pam_issue.so*
-rwxr-xr-x 1 root root 9774 Jun 3 13:46 pam_lastlog.so*
-rwxr-xr-x 1 root root 13591 Jun 3 13:46 pam_limits.so*
-rwxr-xr-x 1 root root 11268 Jun 3 13:46 pam_listfile.so*
-rwxr-xr-x 1 root root 11182 Jun 3 13:46 pam_mail.so*
-rwxr-xr-x 1 root root 5923 Jun 3 13:46 pam_nologin.so*
-rwxr-xr-x 1 root root 5460 Jun 3 13:46 pam_permit.so*
-rwxr-xr-x 1 root root 18226 Jun 3 13:46 pam_pwcheck.so*
-rwxr-xr-x 1 root root 12590 Jun 3 13:46 pam_rhosts_auth.so*
-rwxr-xr-x 1 root root 5551 Jun 3 13:46 pam_rootok.so*
-rwxr-xr-x 1 root root 7239 Jun 3 13:46 pam_securetty.so*
-rwxr-xr-x 1 root root 6551 Jun 3 13:46 pam_shells.so*
-rwxr-xr-x 1 root root 55925 Jun 4 12:00 pam_smb_auth.so*
-rwxr-xr-x 1 root root 12678 Jun 3 13:46 pam_stress.so*
-rwxr-xr-x 1 root root 11170 Jun 3 13:46 pam_tally.so*
-rwxr-xr-x 1 root root 11124 Jun 3 13:46 pam_time.so*
-rwxr-xr-x 1 root root 45703 Jun 3 13:46 pam_unix.so*
-rwxr-xr-x 1 root root 45703 Jun 3 13:46 pam_unix2.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_acct.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_auth.so*
```

```
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_passwd.so*
-rwxr-xr-x 1 root root 45386 Jun 3 13:46 pam_unix_session.so*
-rwxr-xr-x 1 root root 9726 Jun 3 13:46 pam_userdb.so*
-rwxr-xr-x 1 root root 6424 Jun 3 13:46 pam_warn.so*
-rwxr-xr-x 1 root root 7460 Jun 3 13:46 pam_wheel.so*
./sbin:
total 3132
drwxr-xr-x 2 root root 4096 Jun 4 12:35 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
-rwxr-xr-x 1 root root 178256 Jun 3 13:46 choptest*
-rwxr-xr-x 1 root root 184032 Jun 3 13:46 cqtest*
-rwxr-xr-x 1 root root 81096 Jun 3 13:46 dialtest*
-rwxr-xr-x 1 root root 1142128 Jun 4 11:28 ldconfig*
-rwxr-xr-x 1 root root 2868 Jun 3 13:46 lockname*
-rwxr-xr-x 1 root root 3340 Jun 3 13:46 ondelay*
-rwxr-xr-x 1 root root 376796 Jun 3 13:46 pagesend*
-rwxr-xr-x 1 root root 13950 Jun 3 13:46 probemodem*
-rwxr-xr-x 1 root root 9234 Jun 3 13:46 recvstats*
-rwxr-xr-x 1 root root 64480 Jun 3 13:46 sftp-server*
-rwxr-xr-x 1 root root 744412 Jun 3 13:46 sshd*
-rwxr-xr-x 1 root root 30750 Jun 4 11:46 su*
-rwxr-xr-x 1 root root 194632 Jun 3 13:46 tagtest*
-rwxr-xr-x 1 root root 69892 Jun 3 13:46 tsitest*
-rwxr-xr-x 1 root root 43792 Jun 3 13:46 typetest*
./tmp:
total 8
drwxr-xr-x 2 root root 4096 Jun 4 12:32 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
./usr:
total 8
drwxr-xr-x 2 root root 4096 Jun 4 12:16 ./
drwxr-xr-x 9 root root 4096 Jun 5 10:05 ../
lrwxrwxrwx 1 root root 7 Jun 4 12:14 bin -> ../bin//
lrwxrwxrwx 1 root root 7 Jun 4 11:33 lib -> ../lib//
lrwxrwxrwx 1 root root 8 Jun 4 12:13 sbin -> ../sbin//
```

Anhang H

`Chroot`-Umgebung für `Apache`

H.1 Einleitung

Das Programm `chroot` wird häufig dazu benutzt, einen Daemon in einen beschränkten Verzeichnisbaum einzusperren. Sie können es dazu verwenden, um Dienste von anderen abzuschirmen, so dass Sicherheitsprobleme mit einem Softwarepaket den ganzen Server gefährden können. Durch die Verwendung des Skripts `makejail` wird es viel leichter, einen Verzeichnisbaum in einer `chroot`-Umgebung einzurichten und zu aktualisieren.

FIXME: Apache can also be chrooted using `http://www.modsecurity.org` which is available in `libapache-mod-security` (for Apache 1.x) and `libapache2-mod-security` (for Apache 2.x).

H.1.1 Lizenz

Dieses Dokument ist urheberrechtlich von Alexandre Ratti (2002) geschützt. Es steht unter einer doppelten Lizenz, nämlich der GPL Version 2 (GNU General Public License) und der GNU-FDL 1.2 (GNU Free Documentation Licence). Es wurde in dieses Handbuch mit seiner ausdrücklichen Genehmigung aufgenommen. (Siehe auch das Originaldokument (`http://www.gabuzomeu.net/alex/doc/apache/index-en.html`))

H.2 Installation des Servers

Diese Vorgehensweise wurde auf Debian GNU/Linux 3.0 (Woody) mit `makejail` 0.0.4-1 (in Debian/testing) getestet.

- Melden Sie sich als `Root` an und erstellen Sie ein neues Verzeichnis für das Gefängnis:

```
$ mkdir -p /var/chroot/apache
```

- Erstellen Sie einen neuen Nutzer und eine neue Gruppe. Der Apache in der chroot-Umgebung wird als dieser Nutzer und Gruppe laufen, die für nichts anderes auf dem System verwendet wird. In dem Beispiel heißen sowohl Nutzer als auch Gruppe chrapach.

```
$ adduser --home /var/chroot/apache --shell /bin/false \
--no-create-home --system --group chrapach
```

  FIXME: is a new user needed? (Apache already runs as the apache user)

- Installieren Sie ganz normal Apache auf Debian: `apt-get install apache`
- Richten Sie Apache ein (z.B. definieren Sie Ihrer Subdomains usw.). Weisen Sie in der Konfigurationsdatei `/etc/apache/httpd.conf` den Optionen *Group* und *User* chrapach zu. Starten Sie Apache neu und stellen Sie sicher, dass der Server korrekt funktioniert. Danach halten Sie den Server wieder an.
- Installieren Sie makejail (ist fürs Erste in Debian/testing vorhanden). Sie sollten auch wget und lynx installieren, da sie benutzt von makejail werden, um den Server in der chroot-Umgebung zu testen: `apt-get install makejail wget lynx`.
- Kopieren Sie die Beispielkonfigurationsdatei für Apache ins Verzeichnis /etc /makejail:

```
# cp /usr/share/doc/makejail/examples/apache.py /etc/makejail/
```

- Bearbeiten Sie `/etc/makejail/apache.py`. Sie müssen die Optionen *chroot*, *users* und *groups* verändern. Um diese Version von makejail laufen zu lassen, können Sie auch die Option packages hinzufügen. Vergleichen Sie die Makejail-DoKumentation (http://www.floc.net/makejail/current/doc/). Die Konfigurationsdatei könnte beispielsweise so aussehen:

```
chroot="/var/chroot/apache"
testCommandsInsideJail=["/usr/sbin/apachectl start"]
processNames=["apache"]
testCommandsOutsideJail=["wget -r --spider http://localhost/",
                         "lynx --source https://localhost/"]
preserve=["/var/www",
          "/var/log/apache",
          "/dev/log"]
users=["chrapach"]
groups=["chrapach"]
packages=["apache", "apache-common"]
userFiles=["/etc/password",
```

```
            "/etc/shadow"]
groupFiles=["/etc/group",
            "/etc/gshadow"]
forceCopy=["/etc/hosts",
           "/etc/mime.types"]
```

FIXME: some options do not seem to work properly. For instance, `/etc/shadow` and `/etc/gshadow` are not copied, whereas `/etc/password` and `/etc/group` are fully copied instead of being filtered.

- Erstellen Sie den Verzeichnisbaum für chroot: `makejail /etc/makejail/apache.py`.
- Falls `/etc/password` und `/etc/group` vollständig kopiert wurden, geben Sie Folgendes ein, um sie mit gefilterten Fassungen zu ersetzen:

```
$ grep chrapach /etc/passwd > /var/chroot/apache/etc/passwd
$ grep chrapach /etc/group > /var/chroot/apache/etc/group
```

- Kopieren Sie die Webseiten und die Logs ins Gefängnis. Diese Dateien werden nicht automatisch mitkopiert (sehen Sie sich dazu die Option *preserve* in der Konfigurationsdatei von `makejail` an).

```
# cp -Rp /var/www /var/chroot/apache/var
# cp -Rp /var/log/apache/*.log /var/chroot/apache/var/log/apache
```

- Editieren Sie das Startskript für den Logging-Daemon des Systems so, dass er auch den Socket `/var/chroot/apache/dev/log` beobachtet. Ersetzen Sie in `/etc/init.d/sysklogd SYSLOGD=""` mit `SYSLOGD=" -a /var/chroot/apache/dev/log"` und starten Sie den Daemon neu (`/etc/init.d/sysklogd restart`).
- Editieren Sie das Startskript von Apache (`/etc/init.d/apache`). Sie müssen vielleicht ein paar Änderung am Standardstartskript vornehmen, damit des richtig in einem Verzeichnisbaum in einer `chroot`-Umgebung läuft. Da wäre:
  - Legen Sie die Variable *CHRDIR* am Anfang der Datei neu fest.
  - Bearbeiten Sie die Abschnitte *start*, *stop*, *reload* etc.
  - Fügen Sie eine Zeile hinzu, um das `/proc`-Dateisystem innerhalb des Gefängnisses zu mounten und abzumounten.

```
#! /bin/bash
#
# apache        Start the apache HTTP server.
#

CHRDIR=/var/chroot/apache
```

```
NAME=apache
PATH=/bin:/usr/bin:/sbin:/usr/sbin
DAEMON=/usr/sbin/apache
SUEXEC=/usr/lib/apache/suexec
PIDFILE=/var/run/$NAME.pid
CONF=/etc/apache/httpd.conf
APACHECTL=/usr/sbin/apachectl

trap "" 1
export LANG=C
export PATH

test -f $DAEMON || exit 0
test -f $APACHECTL || exit 0

# ensure we don't leak environment vars into apachectl
APACHECTL="env -i LANG=${LANG} PATH=${PATH} chroot $CHRDIR $APACHECTL"

if egrep -q -i "^[[:space:]]*ServerType[[:space:]]+inet" $CONF
then
    exit 0
fi

case "$1" in
  start)
    echo -n "Starting web server: $NAME"
    mount -t proc proc /var/chroot/apache/proc
    start-stop-daemon --start --pidfile $PIDFILE --exec $DAEMON \
      --chroot $CHRDIR
    ;;

  stop)
    echo -n "Stopping web server: $NAME"
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" --oknodo
    umount /var/chroot/apache/proc
    ;;

  reload)
    echo -n "Reloading $NAME configuration"
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" \
      --signal USR1 --startas $DAEMON --chroot $CHRDIR
    ;;

  reload-modules)
    echo -n "Reloading $NAME modules"
```

```
    start-stop-daemon --stop --pidfile "$CHRDIR/$PIDFILE" --oknodo \
      --retry 30
    start-stop-daemon --start --pidfile $PIDFILE \
      --exec $DAEMON --chroot $CHRDIR
    ;;

  restart)
    $0 reload-modules
    exit $?
    ;;

  force-reload)
    $0 reload-modules
    exit $?
    ;;

  *)
    echo "Usage: /etc/init.d/$NAME {start|stop|reload|reload-modules|fo
    exit 1
    ;;
esac

if [ $? == 0 ]; then
  echo .
  exit 0
else
  echo failed
  exit 1
fi
```

FIXME: should the first Apache process be run as another user than root (i.e. add –chuid chrapach:chrapach)? Cons: chrapach will need write access to the logs, which is awkward.

- Ersetzen Sie in `/etc/logrotate.d/apache` `/var/log/apache/*.log` durch `/var/chroot/apache/var/log/apache/*.log`.
- Starten Sie Apache (`/etc/init.d/apache start`) und überprüfen Sie, was im Protokoll des Gefängnisses gemeldet wird (`/var/chroot/apache/var/log/apache/error.log`). Wenn Ihre Einstellung komplexer sein sollte (z.B. wenn Sie auch PHP und MySQL einsetzen), werden wahrscheinlich Dateien fehlen. Wenn einige Dateien nicht automatisch von `makejail` kopiert werden, können Sie diese in den Optionen *forceCopy* (um Dateien direkt zu kopieren) oder *packages* (um ganze Pakete mit ihren Abhängigkeiten zu kopieren) in der Konfigurationsdatei `/etc/makejail/apache.py` aufführen.
- Geben Sie `ps aux | grep apache` ein, um sicherzustellen, dass Apache läuft. Sie sollten etwas in dieser Art sehen:

```
root 180 0.0 1.1 2936 1436 ? S 04:03 0:00 /usr/sbin/apache
chrapach 189 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 190 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 191 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 192 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
chrapach 193 0.0 1.1 2960 1456 ? S 04:03 0:00 /usr/sbin/apache
```

- Stellen Sie sicher, dass die Apache-Prozesse in einer `chroot`-Umgebung laufen. Betrachten Sie dazu das `/proc`-Dateisystem: `ls -la /proc/`*`process_number`*`/root/.`, wobei *process_number* einer der PID-Nummern ist, die oben aufgeführt wurden (z.B. 189 in der zweiten Reihe). Die Einträge des eingeschränkten Verzeichnisbaums sollten Sie sich auflisten lassen:

  ```
  drwxr-sr-x 10 root staff 240 Dec 2 16:06 .
  drwxrwsr-x 4 root staff 72 Dec 2 08:07 ..
  drwxr-xr-x 2 root root 144 Dec 2 16:05 bin
  drwxr-xr-x 2 root root 120 Dec 3 04:03 dev
  drwxr-xr-x 5 root root 408 Dec 3 04:03 etc
  drwxr-xr-x 2 root root 800 Dec 2 16:06 lib
  dr-xr-xr-x 43 root root 0 Dec 3 05:03 proc
  drwxr-xr-x 2 root root 48 Dec 2 16:06 sbin
  drwxr-xr-x 6 root root 144 Dec 2 16:04 usr
  drwxr-xr-x 7 root root 168 Dec 2 16:06 var
  ```

  Um diesen Test zu automatisieren, geben Sie `ls -la /proc/‘cat /var/chroot/apache/var/run/apache.pid‘/root/.` ein.

  FIXME: Add other tests that can be run to make sure the jail is closed?

Ich mag das, da es so nicht sehr schwierig ist, das Gefängnis einzurichten, und der Server mit nur zwei Zeilen aktualisiert werden kann:

```
apt-get update && apt-get install apache
makejail /etc/makejail/apache.py
```

H.3 Weiterführende Informationen

Wenn Sie nach weiteren Informationen suchen, sehen Sie sich die Quellen an, auf denen diese Anleitung beruht:

- Die Makejail-Homepage (http://www.floc.net/makejail/). Diese Programm wurde von Alain Tesio geschrieben.

- Das Chrooting daemons and system processes HOWTO (http://www.nuclearelephant.com/papers/chroot.html) von Jonathan, Network Dweebs, 21.10.2002.