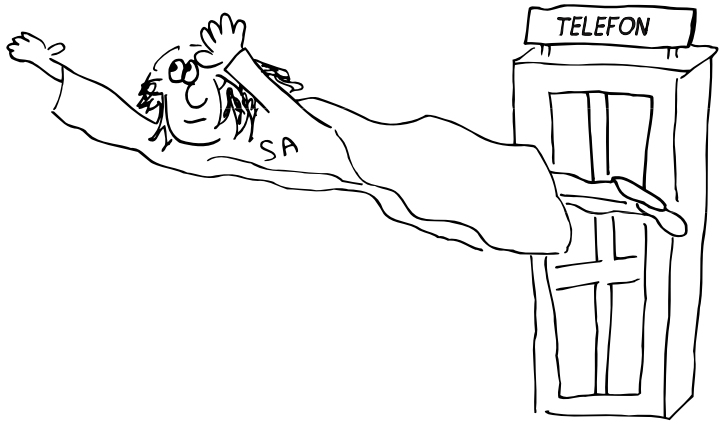
Kapitel 3

root-Power



Jede Datei und jeder Prozess auf einem Linux-System sind einem bestimmen Benutzer-Account zugeordnet. Ohne Erlaubnis des Eigentümers können andere Benutzer nicht auf diese Objekte zugreifen. Diese Konvention hilft also, uns gegen die Missetaten der anderen zu schützen, seien sie nun beabsichtigt oder versehentlich.

Systemdateien und -prozesse gehören einem fiktiven Benutzer, »root«, auch als Superuser bezeichnet. Wie bei den anderen Accounts ist auch das Eigentum von root gegen Störungen durch andere Benutzer geschützt. Um administrative Änderungen vorzunehmen, verwenden Sie eine der Methoden für den Zugriff auf den root-Account, die in diesem Kapitel beschrieben werden.

Der root-Account hat mehrere »magische« Eigenschaften. root kann als Eigentümer jeder Datei und jedes Prozesses auftreten. Außerdem kann root verschiedene spezielle Operationen durchführen, die anderen Benutzern nicht erlaubt sind. Der Account ist äußerst leistungsfähig und kann in unwissenden oder böswilligen Händen äußerst gefährlich sein.

Dieses Kapitel führt in die Grundlagen des Superuser-Zugriffs für Administratoren ein. *Kapitel 21* beschreibt, wie man unerwünschten und störenden Superuser-Zugriff durch andere Anwender verhindert. *Kapitel 29* beschäftigt sich mit wichtigen strategischen und administrativen Aspekten.

3.1 Eigentum an Dateien und Prozessen

Jede Datei hat sowohl einen Eigentümer als auch einen »Gruppeneintrag«. Der Eigentümer einer Datei besitzt nur ein Privileg, das sonst niemand auf dem System hat: die Möglichkeit, die Zugriffsberechtigung für die Datei zu ändern. Insbesondere kann der Eigentümer diese Zugriffsberechtigung so restriktiv gestalten, dass kein anderer darauf zugreifen kann[1]. Wir werden in *Kapitel 5* noch einmal auf das Thema der Dateiberechtigungen zurückkommen.

> **VERWEIS**
> Weitere Informationen über Gruppen finden Sie in *Kapitel 6*.

Obwohl es sich beim Eigentümer einer Datei immer um eine einzelne Person handelt, steuert der Gruppeneintrag die Zugriffsrechte für eine ganze Gruppe von Benutzern, wenn sie alle Mitglied einer bestimmten Unix-Gruppe sind. Gruppen sind in der Datei `/etc/group` definiert.

Der Eigentümer einer Datei bestimmt, welche Operationen die Gruppenmitglieder dafür ausführen dürfen. Dieses Schema erlaubt, dass Mitglieder desselben Projekts eine Datei gemeinsam nutzen. Wir verwenden beispielsweise eine Gruppe, um den Zugriff auf die Quelldateien für die Website `www.admin.com` zu steuern.

Beide Eigentumsverhältnisse einer Datei werden mithilfe von `ls -lg` *Dateiname* ermittelt, z.B. wie folgt:

```
% ls -lg /staff/scott/todo
-rw------- 1 scott staff 1258 Jun 4 18:15 /staff/scott/todo
```

Diese Datei gehört »scott« und steht der Gruppe »staff« zur Verfügung.

Linux verwaltet die Namen von Eigentümern und Gruppen als Nummern, nicht als Text. Die Benutzernummern (User Identification Numbers) kurz UIDs) werden in der Datei `/etc/passwd` den Benutzernamen zugeordnet, die Gruppennummern (Group Identification Numbers, GIDs) werden in `/etc/group` auf die entsprechenden Gruppennamen abgebildet[2]. Die Textnamen, die den UIDs und GIDs entsprechen, werden in den meisten Situationen nur der besseren Lesbarkeit für den Benutzer halber bereitgestellt. Wenn Befehle wie beispielsweise `ls` Eigentümerinformationen in einem vom Menschen lesbaren Format darstellen wollen, müssen sie diese Namen in der entsprechenden Datei oder Datenbank nachschlagen.

Prozessen sind nicht nur zwei, sondern vier IDs zugeordnet: eine reale und eine effektive UID, sowie eine reale und eine effektive GID. Die »realen« Nummern werden für das Accounting verwendet, die »effektiven« Nummern helfen, die Zugriffsberechtigungen zu ermitteln. Normalerweise sind die reale und die effektive Nummer gleich. Der Eigentümer eines Prozesses kann Prozesssignale senden (siehe *Abschnitt 5.4*) und außerdem die Scheduling-Priorität des Prozesses verringern (herabstufen).

---

1 Tatsächlich können die Zugriffsberechtigungen so restriktiv gesetzt werden, dass selbst der Eigentümer einer Datei keinen Zugriff mehr darauf hat – eine Funktionalität, die sehr viel praktischer ist, als es auf den ersten Blick den Anschein haben mag.

2 Einige Systeme speichern diese Information nicht mehr in Textdateien. Weitere Informationen finden Sie in *Kapitel 18*.

VERWEIS
Weitere Informationen über die Berechtigungsbits finden Sie ab *Abschnitt 5.5*.

Obwohl es für einen Prozess normalerweise nicht möglich ist, die vier Eigentumsreferenzen zu ändern, gibt es eine spezielle Situation, in der effektive Benutzer- und Gruppen-ID geändert werden können. Wird ein Befehl ausgeführt, dessen Berechtigungsbit »setuid« oder »setgid« gesetzt ist, können die effektive UID und GID des Prozesses auf die UID und GID der Datei gesetzt werden, die das neue Programmabbild enthält, statt auf die UID und die GID des Benutzers, der den Befehl ausgeführt hat. Die Benutzerrechte werden also nur für die Ausführung dieses speziellen Befehls »weitergegeben«.

Die setuid-Funktionalität von Linux erlaubt Programmen, die von normalen Benutzern ausgeführt werden, den root-Account auf begrenzte und streng kontrollierte Weise zu nutzen. Beispielsweise ist der Befehl `passwd`, den Benutzer ausführen, um ihr Anmeldepasswort zu ändern, ein setuid-Programm. Es ändert die Datei `/etc/shadow` (oder die Datei `/etc/passwd`) auf wohldefinierte Weise ab und wird dann beendet. Natürlich bietet selbst diese beschränkte Nutzung die Möglichkeit eines Missbrauchs, deshalb fordert `passwd` von den Benutzern, zu beweisen, dass sie das aktuelle Passwort kennen, bevor sie die gewünschte Änderung vornehmen.

3.2 Der Superuser

Die charakteristische Eigenschaft des root-Accounts ist die UID 0. Linux erlaubt, den Benutzernamen dieses Accounts zu ändern oder zusätzliche Accounts mit der UID 0 anzulegen, aber beides ist nicht sehr sinnvoll. Solche Änderungen unterstützen tendenziell unerwünschte Gefährdungen der Systemsicherheit. Außerdem verursachen sie Verwirrung, wenn andere Leute sich mit der eigentümlichen Konfiguration Ihres Systems auseinandersetzen müssen.

Linux erlaubt dem Superuser (eigentlich jedem Prozess mit der effektiven UID 0), alle gültigen Operationen für eine Datei oder einen Prozess auszuführen[3]. Darüber hinaus können einige Systemaufrufe (Anforderungen an den Kernel) nur vom Superuser ausgeführt werden. Einige Beispiele für eingeschränkte Operationen sind:

- Wechseln des Stammverzeichnisses eines Prozesses mit `chroot`
- Gerätedateien erzeugen
- Setzen der Systemuhr
- Ressourcen-Beschränkungen erhöhen und Prozessprioritäten setzen
- den Hostnamen des Systems festlegen
- Netzwerkschnittstellen konfigurieren
- Privilegierte Netzwerkports öffnen (deren Nummern über 1024 liegen)
- das System herunterfahren

[3] »Gültig« ist hier ein wichtiges Schlüsselwort. Bestimmte Operationen (wie beispielsweise die Ausführung einer Datei, deren Ausführungsberechtigungsbit nicht gesetzt ist) sind sogar für den Superuser verboten.

Ein Beispiel für die herausragende Stellung des Superusers ist, dass ein Prozess, der root gehört, auch seine UID und GID beliebig ändern kann. Ein Beispiel für einen solchen Prozess ist das Programm `login` und dessen Gegenstücke im Window-System; der Prozess, der Sie auffordert, Ihr Passwort einzugeben, wenn Sie sich am System anmelden, wird zunächst als root ausgeführt. Wenn der eingegebene Name und das Passwort korrekt sind, ändert das Programm `login` seine UID und GID auf Ihre UID und GID und startet Ihre Benutzerumgebung. Wenn ein root-Prozess einmal seine Eigentümerrechte geändert hat und zu einem normalen Anwenderprozess geworden ist, kann er seinen zuvor privilegierten Status nicht mehr zurückerhalten.

3.3 Ein root-Passwort auswählen

VERWEIS
Weitere Informationen über das Knacken von Passwörtern finden Sie ab *Abschnitt 21.8*.

Das root-Passwort sollte mindestens 8 Zeichen lang sein. Passwörter mit sieben Zeichen können in der Regel wesentlich einfacher geknackt werden. Auf Systemen, die DES-Passworte verwenden, ist es nicht sinnvoll, ein Passwort mit mehr als acht Zeichen zu verwenden, weil nur die ersten acht Zeichen berücksichtigt werden. Im *Abschnitt 6.1.2* erhalten Sie Informationen darüber, wie Sie MD5-Passworte aktivieren können, deren Länge mehr als 8 Zeichen betragen kann.

Es ist wichtig, das root-Passwort so zu wählen, dass es nicht einfach erraten oder durch Ausprobieren herausgefunden werden kann. Theoretisch setzt sich ein sicheres Passwort aus einer beliebigen Abfolge von Buchstaben, Interpunktionszeichen und Ziffern zusammen. Diese Passwörter kann man sich jedoch nur schwer merken, in der Regel sind sie auch schwierig zu tippen, und sie sind auch nicht mehr sicher, wenn der Systemadministrator sie sich aufschreiben muss oder sie langsam tippt.

Bis vor kurzem stellte ein Passwort aus zwei zufällig gewählten Wörtern getrennt durch ein Interpunktionszeichen einen guten Kompromiss zwischen Sicherheit und Handhabbarkeit dar. Leider können solche Passwörter mittlerweile ziemlich schnell geknackt werden; wir raten daher von diesem Schema ab.

Heute könnten Sie beispielsweise ein sicheres root-Passwort erzeugen, indem Sie sich einen Satz mit »schockierendem Unsinn« ausdenken, wie Grady Ward es in einer früheren Version der PGP Passphrase FAQ ausgedrückt hat:

> *»Schockierender Unsinn« bedeutet, sich einen kurzen Satz auszudenken, der in der Kultur des Benutzers sowohl unsinnig als auch schockierend ist. Das bedeutet, er enthält stark obszöne, rassistische, unmögliche oder anderweitig extreme Gedankengänge. Diese Technik ist zulässig, weil der Satz als solcher niemals jemandem zu Gesicht kommt, dessen Gefühle er verletzen könnte.*
>
> *Schockierender Unsinn kann sehr wahrscheinlich nicht von jemand anderem nachgebildet werden, weil er keine Tatsachen reflektiert, die versehentlich von jemand anderem entdeckt werden könnten. Der emotionale Einfluss bewirkt, dass der Erschaffer dieses Satzes ihn sehr wahrscheinlich nicht vergisst. Ein gemäßigtes Beispiel für derartig schockierenden Unsinn*

könnte sein: »Der Chef zwickt mich in meinen freudig galoppierenden Hintern«. Dem Leser fallen zweifellos sehr viel schockierendere oder unterhaltsamere Beispiele ein.

Sie reduzieren einen solchen Satz auf ein Passwort, indem Sie jeweils nur den ersten Buchstaben jedes Worts verwenden. Sie können aber auch beliebige andere Umwandlungen vornehmen. Die Passwortsicherheit wird extrem verbessert, wenn Sie auch Ziffern, Interpunktionszeichen und Großbuchstaben verwenden.

Wann sollten Sie das root-Passwort ändern?

- mindestens alle drei Monate
- immer wenn jemand, der das Passwort kennt, Ihre Firma verlässt
- wenn Sie denken, dass die Sicherheit nicht mehr gewährleistet ist
- möglichst nur an einem Tag, an dem Sie abends nicht so lange ausgehen, dass Sie das Passwort am nächsten Morgen wieder vergessen haben

3.4 Wie man zum root-Anwender wird

Weil root auch ein Benutzer ist, könnten Sie sich direkt am root-Account anmelden. Das ist jedoch nicht besonders sinnvoll. Erstens werden keine Aufzeichnungen darüber geführt, was Sie als root gemacht haben. Das erweist sich als fatal, wenn Sie nachts um drei merken, dass Sie irgendetwas zerstört haben, sich aber nicht merken konnten, was Sie geändert haben; und es ist noch schlimmer, wenn ein Zugriff unberechtigt war und Sie herausfinden wollen, was der Eindringling in Ihrem System gemacht hat. Ein weiterer Nachteil ist, dass bei der Anmeldung als root keine Aufzeichnungen darüber erfolgen, wer die Arbeiten ausgeführt hat. Wenn mehrere Leute Zugriff auf den root-Account haben, erkennen Sie nicht, wer ihn wann verwendet hat.

Aus diesen Gründen erlauben die meisten Systeme, root-Anmeldungen auf Terminals und über das Netzwerk zu deaktivieren, sodass man sich nur noch von der Konsole aus anmelden kann. Wir empfehlen Ihnen, diese Möglichkeit zu nutzen. Weitere Informationen darüber, welche Datei Sie für Ihr jeweiliges System anpassen müssen, finden Sie ab *Abschnitt 21.7*.

3.4.1 su: Die Benutzer-ID ersetzen

Eine etwas bessere Methode, auf den root-Account zuzugreifen, ist die Verwendung des Befehls `su`. Beim Aufruf ohne Argumente fragt `su` nach dem root-Passwort und startet dann eine root-Shell. Die Berechtigungen dieser Shell bleiben in Kraft, bis die Shell beendet wird (mit Strg+D oder mit dem Befehl `exit`). `su` zeichnet die als root ausgeführten Befehle nicht auf, erzeugt aber einen Protokolleintrag, der angibt, wer wann root genutzt hat.

Der Befehl `su` kann auch andere IDs als root ersetzen. Manchmal ist die einzige Möglichkeit, das Problem eines Benutzers zu reproduzieren oder zu debuggen, ein `su` für seinen Account anzulegen, sodass Sie in die Umgebung gelangen, in der das Problem auftritt.

Wenn Sie das Passwort eines Benutzers kennen, erhalten Sie durch Ausführung von `su` *benutzername* direkten Zugriff auf seinen Account. Wie bei einem `su` für root werden Sie nach dem Passwort für *benutzername* gefragt. Sie können auch zuerst `su` auf root und dann auf einen anderen Account ausführen. root kann `su` für jeden anderen Account ausführen, ohne ein Passwort angeben zu müssen.

Sie sollten sich angewöhnen, den vollständigen Pfadnamen für `su` anzugeben (zum Beispiel `/bin/su`), statt sich darauf zu verlassen, dass die Shell den Befehl für Sie finden wird. Damit sind Sie gegenüber Programmen mit dem Namen `su` geschützt, die man möglicherweise in Ihren Suchpfad eingeschmuggelt hat, um Passwörter auszuspionieren[4].

3.4.2 sudo: Ein eingeschränktes su

Weil die Privilegien des Superuser-Accounts nicht unterteilt werden können, ist es schwierig, jemandem das Recht zu erteilen, eine bestimmte Aufgabe zu übernehmen (wie beispielsweise Datensicherungen anzulegen), ohne ihm gleichzeitig freie Bewegung im System zuzugestehen. Und wenn der root-Account von mehreren Administratoren verwendet wird, können Sie kaum kontrollieren, wer ihn verwendet oder was damit gemacht wurde.

Unsere Lösung für diese Probleme ist das Programm `sudo`, das momentan von Todd Miller gewartet wird. Es wurde als Paket von Red Hat, SuSE und Debian organisiert, aber Sie finden den Quellcode auch unter der Internetadresse `www.courtesan.com`.

`sudo` nimmt als Argument eine Befehlszeile entgegen, die als root ausgeführt werden soll (oder als beschränkter Benutzer). `sudo` wertet die Datei `/etc/sudoers` aus, die eine Liste der Benutzer enthält, welche die Berechtigung für `sudo` besitzen, sowie der Befehle, die diese auf einem bestimmten Host ausführen dürfen. Wenn der angeforderte Befehl erlaubt ist, fordert `sudo` zur Eingabe des *eigenen Passworts des Benutzers* auf und führt den Befehl aus.

Weitere `sudo`-Befehle können ohne die erneute Eingabe eines Passworts durch den aktuellen Benutzer ausgeführt werden, falls nicht mehr als fünf Minuten (oder ein anderes einstellbares Zeitintervall) seit der letzten `sudo`-Aktivität vergangen sind. Dieser Timeout dient als einfacher Schutz gegen Benutzer mit `sudo`-Privilegien, die ihre Terminals unbeaufsichtigt zurücklassen.

`sudo` erstellt außerdem ein Protokoll der ausgeführten Befehle, der Anwender, die sie angefordert haben, der Verzeichnisse, von denen aus sie ausgeführt wurden, und der Aufrufzeitpunkte. Diese Information wird mithilfe von `syslog` aufgezeichnet und kann in eine beliebige Datei geschrieben werden. Wir empfehlen Ihnen, die Protokolleinträge mit `syslog` an einen »sicheren« zentralen Host weiterzuleiten.

---

[4] Aus demselben Grund empfehlen wir Ihnen sehr, nicht das aktuelle Verzeichnis ».« in den Suchpfad Ihrer Shell aufzunehmen. Diese Konfiguration ist zwar praktisch, aber gleichzeitig besteht damit die Gefahr, dass unbeabsichtigt »spezielle« Versionen von Systembefehlen ausgeführt werden, die ein Benutzer oder ein Eindringling hinterlassen hat, um eine Falle zu stellen. Diese Empfehlung gilt natürlich besonders für root.

Ein Protokolleintrag für eine Ausführung von `sudo /bin/cat/etc/sudoers` durch randy könnte wie folgt aussehen:

```
Dec 7 10:57:19 tigger sudo: randy: TTY=ttyp0; PWD=/tigger/users/randy;
   USER=root ; COMMAND=/bin/cat /etc/sudoers
```

Die Datei `sudoers` ist so ausgelegt, dass eine einzige Version von mehreren verschiedenen Hosts gleichzeitig genutzt werden kann. Hier ein typisches Beispiel:

```
# Aliase für Maschinen der Informatik- und Physik-
# Lehrstühle erzeugen
Host_Alias     CS = tigger, anchor, piper, moet, sigi
Host_Alias     PHYSICS = eprince, pprince, icarus

# Befehlsauflistungen definieren
Cmnd_Alias     DUMP = /sbin/dump, /sbin/restore
Cmnd_Alias     PRINTING = /usr/sbin/lpc, /usr/bin/lprm
Cmnd_Alias     SHELLS = /bin/sh, /bin/tcsh, /bin/csh, /bin/bash, /bin/bsh,

# Berechtigungen
mark, ed     PHYSICS = ALL
herb         CS = /usr/local/bin/tcpdump : PHYSICS = (operator) DUMP
lynda        ALL = (ALL) ALL, !SHELLS
%wheel       ALL, !PHYSICS = NOPASSWD: PRINTING
```

Die ersten fünf nicht auskommentierten Zeilen definieren Gruppen von Hosts und Befehlen, auf die in den Berechtigungen später in der Datei verwiesen wird. Die Listen könnten auch direkt in diese Spezifikationen aufgenommen werden, aber die Verwendung von Aliasen bewirkt, dass die Datei `sudoers` einfacher zu lesen und leichter zu verstehen ist. Außerdem kann die Datei dadurch später einfacher aktualisiert werden. Darüber hinaus ist es möglich, Aliase für bestimmte Benutzer sowie für Benutzer, für die bestimmte Befehle ausgeführt werden können, zu definieren.

Jede Zeile mit einer Spezifikation von Berechtigungen enthält die folgenden Informationen:

- die Benutzer, für die die Zeile gilt
- die Hosts, auf denen die Zeile angelegt werden soll
- die Befehle, die die angegebenen Benutzer ausführen dürfen
- die Benutzer, für die die Befehle ausgeführt werden dürfen

Die erste Berechtigungszeile gilt für die Benutzer mark und ed auf den Maschinen in der Gruppe `PHYSICS` (eprince, pprince und icarus). Der eingebaute Befehls-Alias `ALL` erlaubt Ihnen, beliebige Befehle auszuführen. Weil keine Benutzerliste in Klammern angegeben ist, führt `sudo` nur Befehle als root aus.

Die zweite Berechtigungszeile erlaubt herb, `tcpdump` auf `CS`-Maschinen und sicherungsrelevante Befehle auf `PHYSICS`-Maschinen auszuführen. Die Dump-Befehle können jedoch nur als operator, nicht als root ausgeführt werden. Die Befehlszeile, die herb demnach eingibt, könnte beispielsweise wie folgt aussehen:

```
% sudo -u operator /usr/sbin/dump 0u /dev/hda2
```

Der Benutzer lynda kann Befehle als beliebiger Benutzer auf jeder Maschine ausführen, bis auf einige allgemeine Shells. Bedeutet das, dass lynda keine root-Shell erhält? Natürlich nicht:

```
% cp -p /bin/csh /tmp/csh
% sudo /tmp/csh
```

Allgemein ausgedrückt bedeutet das, jeder Versuch, »alle Befehle außer ...« zu erlauben, ist zumindest in technischer Hinsicht zum Scheitern verurteilt. Es kann jedoch dennoch als sinnvoll betrachtet werden, die Datei `sudoers` auf diese Weise einzurichten, als Erinnerung, dass root-Shells nicht für gewöhnliche Aufgaben genutzt werden sollen.

Die letzte Zeile erlaubt Benutzern der Gruppe wheel, `lpc` und `lprm` als root auf allen Maschinen auszuführen, außer auf eprince, pprince und icarus. Darüber hinaus ist für die Ausführung der Befehle keine Eingabe des Passworts erforderlich.

Beachten Sie, dass Befehle in `/etc/sudoers` mit dem vollständigen Pfadnamen angegeben werden, um zu verhindern, dass Leute ihre eigenen Programme und Skripten als root ausführen. Es sind hier zwar keine Beispiele aufgeführt, aber es kann auch festgelegt werden, mit welchen Argumenten die erlaubten Befehle ausgeführt werden dürfen. Die hier dargestellte einfache Konfiguration zeigt nur einen kleinen Teil der Vielfältigkeit und Eleganz, die die Datei `sudoers` bereitstellt.

Um `/etc/sudoers` zu bearbeiten, verwenden Sie den Befehl `visudo`, der sicherstellt, dass niemand anderer die Datei bearbeitet, einen Editor dafür aufruft und dann die Syntax der bearbeiteten Datei überprüft, bevor sie installiert wird. Dieser letzte Schritt ist besonders wichtig, weil eine ungültige `sudoers`-Datei verhindern könnte, dass Sie `sudo` erneut ausführen, um sie zu korrigieren.

Der Einsatz von `sudo` hat folgende Vorteile:

- Durch die Befehlsprotokollierung wird ein sehr viel besseres Accounting gewährleistet.
- Operatoren können ihre Routinearbeiten mit uneingeschränkten root-Privilegien ausführen.
- Das eigentliche root-Passwort muss nur ein oder zwei Anwendern bekannt sein.
- `sudo` ist schneller als `su` oder die Anmeldung als root.
- Privilegien können entzogen werden, ohne dass das root-Passwort geändert werden muss.
- Eine verbindliche Liste aller Anwender mit root-Bibliotheken wird verwaltet.
- Es besteht kaum die Möglichkeit, dass eine root-Shell unbeaufsichtigt zurückbleibt.
- Für die Zugriffssteuerung für das gesamte Netzwerk kann eine einzige Datei eingesetzt werden.

Es gibt aber auch einige Nachteile. Der schlimmste ist, dass jede Sicherheitsverletzung des persönlichen Accounts eines `sudo`-Benutzers der Verletzung des root-Accounts gleichkommt. Sie können nicht viel dagegen tun, außer dass Sie die Teilnehmer an Ihrem `sudo` sorgfältig auswählen und ihre Accounts ähnlich wie das root-

Account schützen. Sie können auch regelmäßig den Befehl `crack` auf die Passworte von `sudoers` absetzen, um sicherzustellen, dass es sich hierbei um gut ausgewählte Passworte handelt.

> **VERWEIS**
> Weitere Informationen über `crack` erhalten Sie in *Abschnitt 21.8.5*.

Darüber hinaus kann das Befehls-Logging von `sudo` durch bestimmte Tricks unterlaufen werden, etwa Shell-Escapes aus einem erlaubten Programm oder durch `sudo csh` und `sudo su`, falls Sie das erlauben.

3.4.3 Andere Pseudo-Benutzer

root ist der einzige Benutzer, der in den Augen des Linux-Kernels einen besonderen Status einnimmt, aber das System definiert mehrere andere Pseudo-Benutzer. Es ist üblich, das verschlüsselte Passwortfeld dieser speziellen Anwender in der Datei `/etc/shadow` durch einen Stern zu ersetzen, sodass sich niemand in ihren Accounts anmelden kann.

3.4.4 bin: Rechtmäßiger Eigentümer von Systembefehlen

Auf einigen älteren Unix-Systemen besaß der Anwender bin die Verzeichnisse mit den Systembefehlen sowie die meisten ausführbaren Befehle selbst. Heutzutage wird dieser Account jedoch häufig als überflüssig erachtet (und zum Teil sogar als unsicher), deshalb verwenden moderne Systeme im Allgemeinen nur den root-Account. Auf der anderen Seite ist nun mal der Account bin ein »Standard« und kann daher nicht einfach abgeschafft werden.

3.4.5 daemon: Eigentümer nicht-privilegierter Software

Dateien und Prozesse, die Teil des Betriebssystems sind, nicht aber Eigentum von root sein müssen, werden manchmal einem Dämon zugeordnet. Man ging theoretisch davon aus, dass man auf diese Weise die Sicherheitsrisiken vermeiden könnte, die mit dem Eigentum durch root verbunden sind. Darüber hinaus gibt es aus ähnlichen Gründen auch eine Gruppe mit der Bezeichnung »daemon«. Genau wie der Account bin wird auch der Account daemon von den meisten Linux-Distributionen nicht sehr häufig verwendet.

3.4.6 nobody: der generische NFS-Benutzer

> **VERWEIS**
> Weitere Informationen über den Account nobody finden Sie in *Kapitel 17*.

NFS (Network File System) verwendet den Account nobody, um zum Zwecke der gemeinsamen Nutzung von Dateien root-Benutzer auf anderen Systemen darzustellen. Um externe roots von ihrer root-Position zu befreien, muss die externe UID 0 auf etwas anderes als die lokale UID 0 abgebildet werden.

Weil der Account nobody einen generischen und mit relativ wenig Rechten ausgestatteten Benutzer darstellt, soll er keine Dateien besitzen. Wenn nobody Dateien besitzt, sind externe roots in der Lage, die Kontrolle darüber zu übernehmen. Deshalb sollte nobody keine Dateien besitzen.

Eine UID -1 oder -2 wurde früher für nobody verwendet und diese Konvention kann bei einige Distributionen immer noch beobachtet werden, bei denen nobody die UID 65534 hat (die 16-bit zwei-komplementäre Version von -2). Andere weisen einfach eine niedrige UID zu, wie beim Anmelden eines beliebigen System-Accounts, was sinnvoller ist.

3.5 Übungen

☆ 1. Verwenden Sie den Befehl `find` mit der Option `-perm`, um fünf setuid-Dateien in Ihrem System zu finden. Erklären Sie für jede Datei, warum der setuid-Mechanismus notwendig ist, damit der Befehl ordnungsgemäß funktioniert.

☆ 2. Erfinden Sie drei Passwort-Sätze mit »schockierendem Unsinn« und behalten Sie diese für sich. Lassen Sie diese drei Passwort-Sätze durch `md5sum` laufen und beobachten Sie das Ergebnis. Warum ist es sicher, die MD5-Ergebnisse freizugeben?

☆ 3. Zählen Sie eine Reihe von Befehlen auf, die den Passworteintrag eines Benutzers ändern können und zeigen Sie, wie Sie dabei Ihre Spuren verwischen können. Nehmen Sie dabei an, dass Sie nur das Recht `sudo` haben (alle Befehle außer Shells oder `su` sind dabei zulässig).

☆ 4. Erstellen Sie zwei Einträge für die Konfigurationsdatei `sudoers`:

   a) Einen Eintrag, der den Benutzern matt, adam und drew erlaubt, den Drucker zu warten, einen Papierstau zu beheben und die Drucker-Dämonen auf dem Printserver neu zu starten.

   b) Einen Eintrag, der den Benutzern drew, smithgr und jimlane erlaubt, Jobs zu beenden und die Rechner im Studentenübungsraum neu zu starten.

☆ 5. Installieren Sie `sudo` und konfigurieren Sie es so, dass es Ihnen E-Mails zusendet, wenn es einen Missbrauch feststellt. Verwenden Sie es, um die Einträge von `sudo` bezüglich der vorherigen Frage mit lokalen Benutzernamen und Rechnernamen zu testen. Prüfen Sie dabei, dass `sudo` ordnungsgemäß an `syslog` protokolliert. Sehen Sie sich die Einträge von `syslog` an, die durch Ihre Tests entstanden sind. (Dazu ist root-Zugriff erforderlich und Sie werden wahrscheinlich auch die Datei `/etc/syslog.conf` anfassen müssen.)