Sicherheit in Unix-Systemen

Markus Krieger

Rechenzentrum Uni Würzburg

krieger@rz.uni-wuerzburg.de

Inhaltsübersicht

- Einführung und Zielsetzung
- Präventive Maßnahmen
  - Lokale Rechnersicherheit
  - Netzwerksicherheit
  - Einschränkung des Root-Accounts
- Erkennung von Vorfällen
- Bearbeiten eines Einbruchs
- Informationsquellen

Einführung

- Ziel der Veranstaltung
- Definition von Sicherheit und Gefahren
- Denkanstöße
- Angreifer, Angriffsmöglichkeiten & Motivationen
- Schritte eines erfolgreichen Angriffs
- Das Hochschulnetz
- Suche nach einer Strategie
- Sicherheit - warum und wieviel

Ziel der Veranstaltung

Was soll erreicht werden?

- Definition und Diskussion der wichtigsten sicherheitsrelevanten Problemstellungen.
- Was verbirgt sich hinter diversen Schlagwörtern?
- Wo sind die wesentlichen Sicherheitslöcher?
- Wie sehen erste Schritte aus, um die Sicherheit zu verbessern?
- Vorstellung von Utilities zur Verbesserung der Systemsicherheit.

Ziel der Veranstaltung

Was wird nicht erreicht?

- Ein ausgereiftes Sicherheitskonzept.
- Konkrete und genau definierte Schritte was zu tun ist.
- Sichere Unix-Rechner.

Praktische Definition von Systemsicherheit

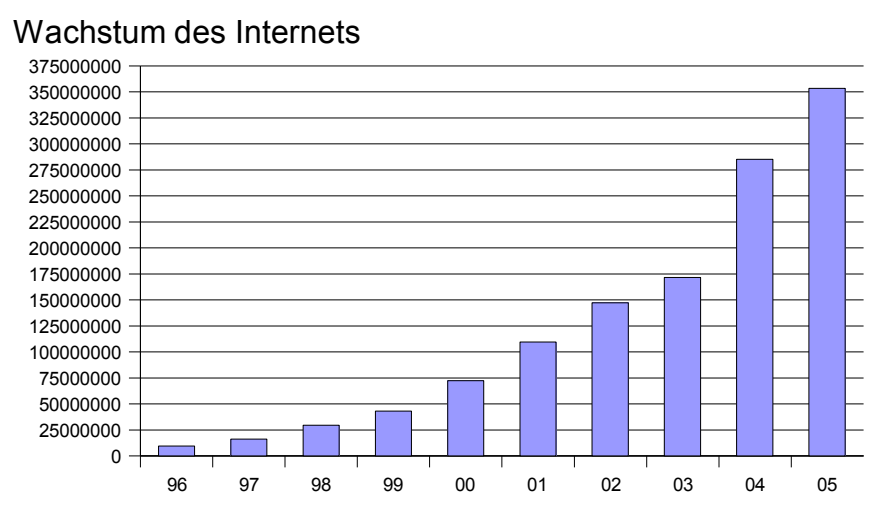
- „Ein System ist sicher, wenn es sich so verhält wie man es von ihm erwartet“
- Sicherheitsansprüche von Benutzern / Administratoren:
  - Integrität von Daten und Programmen
  - Verfügbarkeit
  - Konsistenz
  - Zugriffskontrolle
  - Auditing

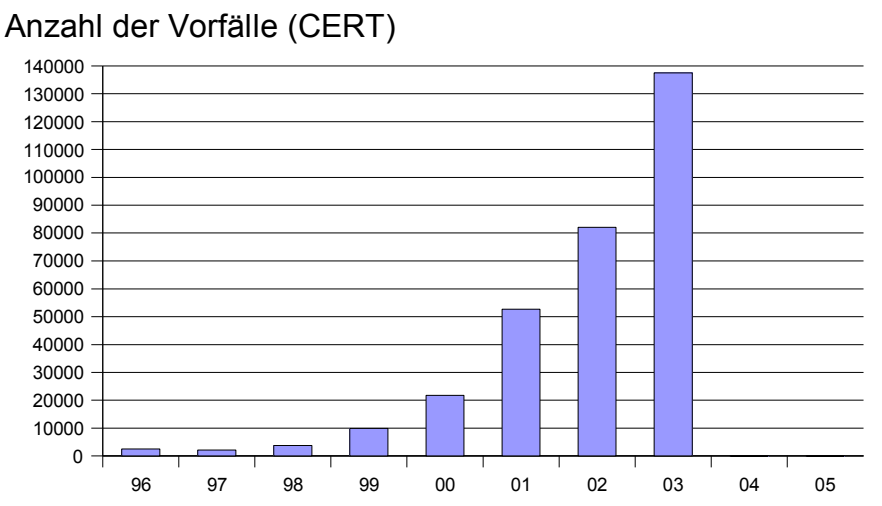
Gefahren für die Sicherheit

- Hardwarefehler
- Softwarefehler
- Fehler von Benutzern / Administratoren
- Äußere Einflüße (Feuer, Diebstahl, ...)
- Angriffe von Innen und von Außen

Dies führt zum Verlust von Integrität, Verfügbarkeit und Vertraulichkeit.

Denkanstöße

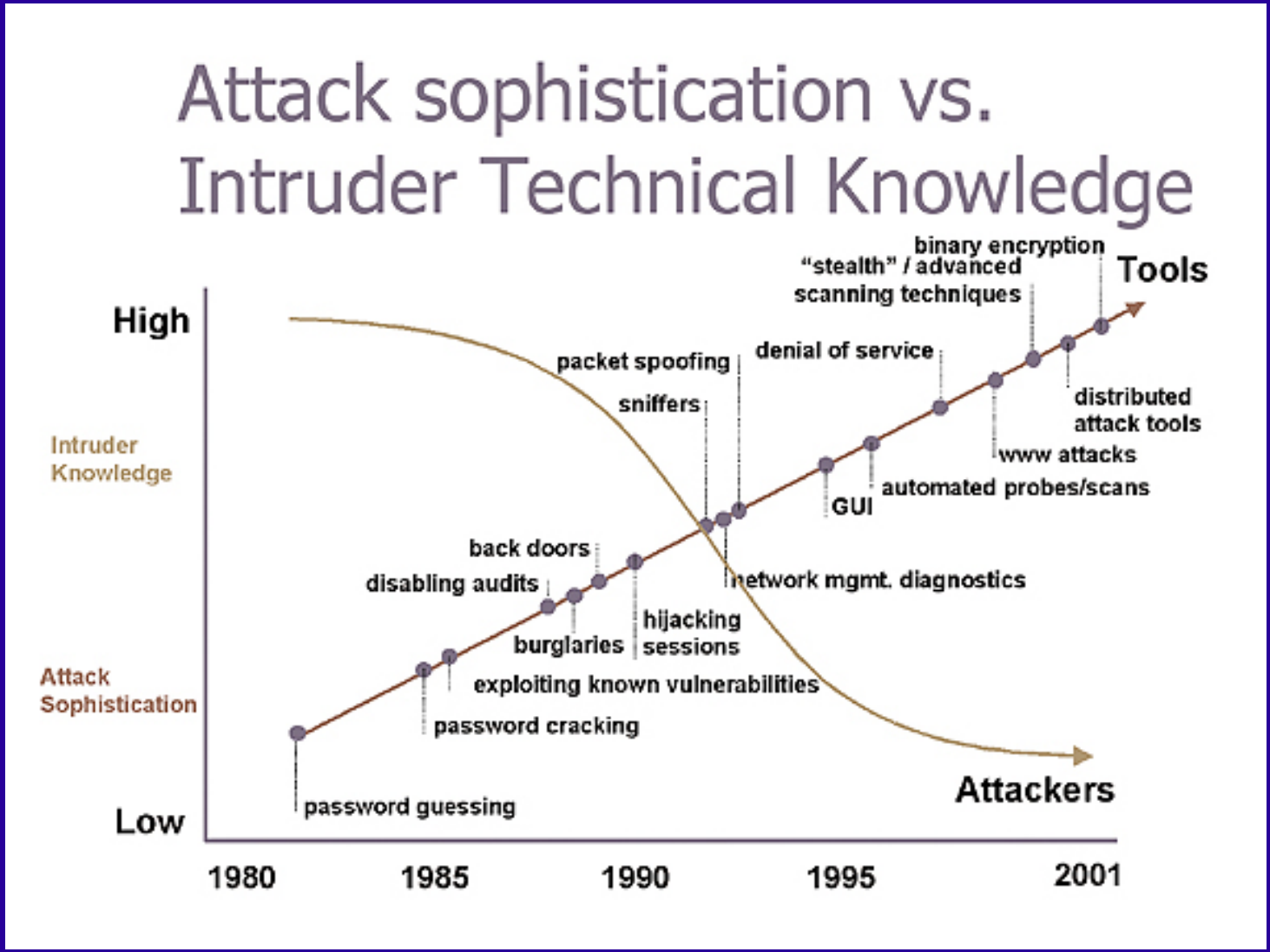




Überlegungen

- Was gibt es für Angriffe?
- Woher kommen die Angriffe?
- Wer greift mich an und warum?
- Wie wichtig sind die Daten meiner Rechner?
- Was kann ich tun und woher bekomme ich Know-how und Informationen?
- Kann ich verantwortlich gemacht werden?

Aufwand eines Angriffs



Angriffsmöglichkeiten

- Lokale / Remote Angriffe
- Trojaner / Viren / Würmer / Backdoors
- Buffer Overflow
- Code Injection
- Falsche Berechtigungen für Dateien / Verzeichnisse
- Fehlerhafte Umgebungsvariablen
- Trust Exploitation (ip-spoofing, hijacking, ...)
- Condition Race
- (D)DOS
- Passives Sniffen

Angreifer

- lokale Angreifer:
  - Studenten
  - Mitarbeiter
  - Erfolgreiche remote Angreifer ohne Root-Privilegien
- remote Angreifer:
  - Kriminelle
  - Hacker
  - Script Kiddies
  - Würmer

Motivation der Angreifer

- Spieltrieb / Neugier
- Zugriff auf Daten / Dienste
- Bandbreite
- Plattenplatz
- CPU für „distributed computing“
- Sprungbrett für weitere Angriffe
- Demonstration von Macht und Können
- Das Hochschulnetz ist offen und die Rechner sind schlecht gesichert

Schritte eines erfolgreichen Angriffs

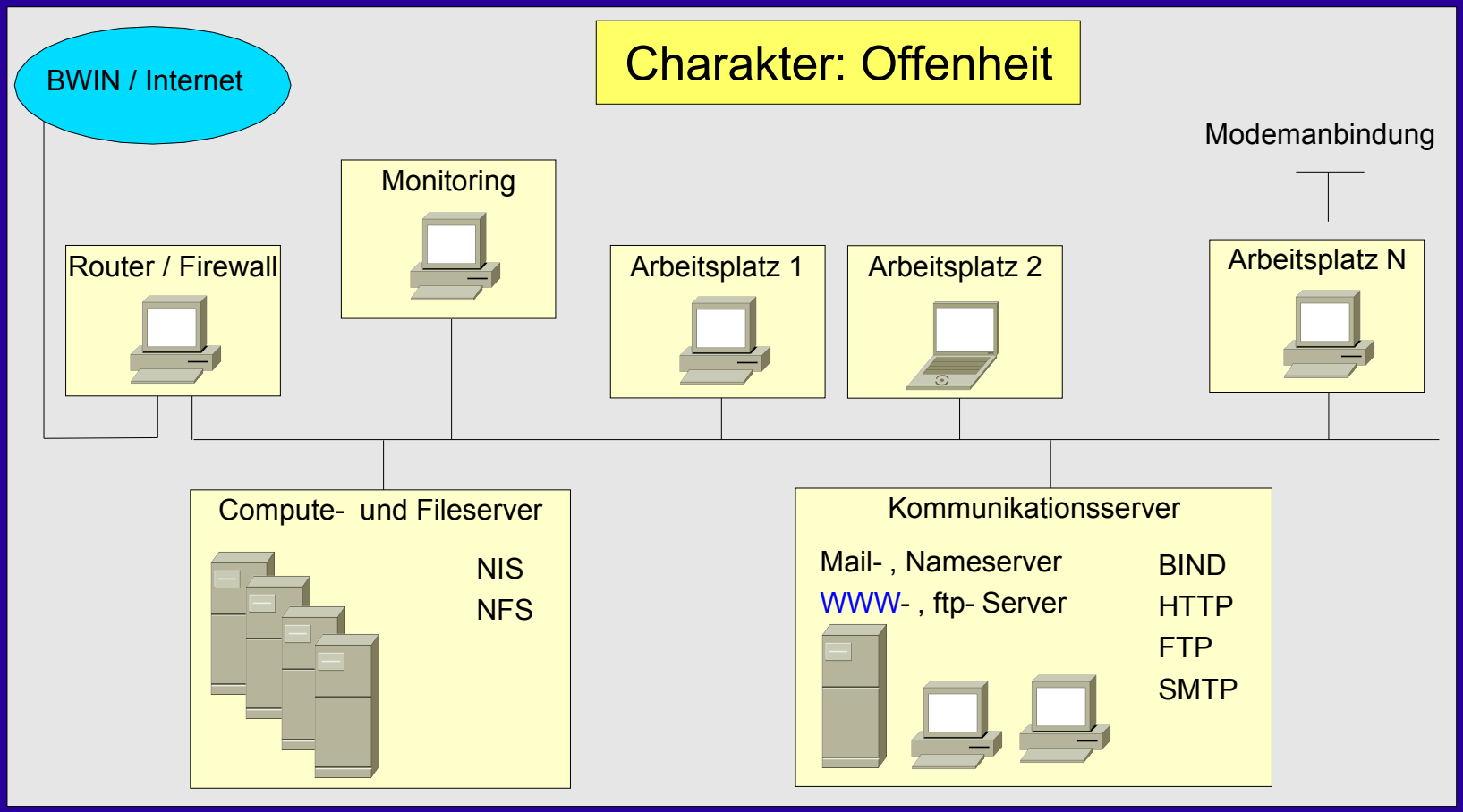
1) Sammeln von Informationen (Scannen nach OS / Diensten, DNS, ...)
2) Informationen nach Verwundbarkeiten durchsuchen
3) Zugang zum System verschaffen
4) Einsatz von exploit-Scripten für root-Rechte
5) Installieren von Backdoors, Sniffern und Rootkits und Verwischen von Spuren

Alternative zu 2) + 3): Ausnutzen von Fehlern in Client-programmen durch manipulierte Webseiten oder Daten (Bilder, Filme, PDF, ...)

Unix-Rechner als beliebte Ziele

- Viele Dienste
- Viele Varianten
- Mächtige Bordmittel
- Bieten die optimale Einbindung ins Internet
- Bieten viele Möglichkeiten der Fehlkonfiguration durch den Administrator
- Laufen oft in einer Defaultinstallation und sind somit angreifbar
- Lange Laufzeiten

Das Hochschulnetz



Suche nach einer Strategie

- Erhöhung der lokalen Sicherheit durch Vorsorgemaßnahmen
  - Physische Sicherheit des Rechners und Netzzuganges
  - Regelmäßige Backups
  - Einspielen aktueller Patches und deaktivieren nicht benötigter Accounts und Dienste
  - Aktivierung von effektivem Logging
  - Weitergabe von Sicherheitsinformationen
  - Einsatz von Tools um die Rechnersicherheit zu überprüfen und um die Grundlage zur Erkennung von Sicherheitsproblemen zu schaffen
  - Einsatz von Kryptographie
  - Führen eines Logbuchs

Suche nach einer Strategie

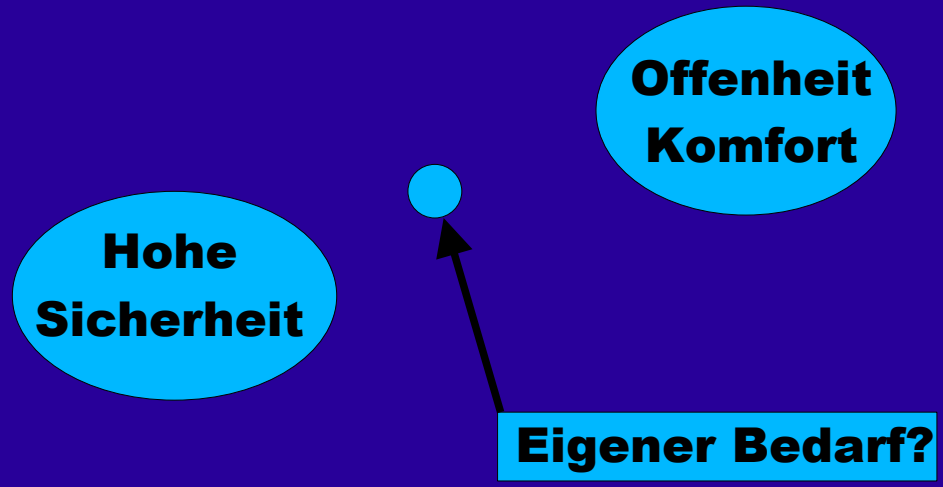
- Erkennen sicherheitsrelevanter Vorfälle
  - Auswerten von Logdateien
  - Regelmäßiges Überprüfen der Rechnerkonfiguration (Dienste und Dateisystem)
  - abnormales Verhalten (Abstürze, hohe Last, ...)
  - externe Informationsquellen (IDS, Meldungen)
- Problematisch dabei:
  - Ist der Rechner gehackt, läßt sich der Zustand mit Bordmitteln nicht mehr verläßlich überprüfen

Suche nach einer Strategie

- Reagieren auf einen Vorfall
  - Ruhe bewaren! Durch voreiliges Handeln kann sonst mehr Schaden als durch den eigentlichen Vorfall angerichtet werden.
  - Befolgen von vorhandenen Sicherheitsrichtlinien
  - Sicherung von Spuren
  - Wiederherstellung des Sollzustands
  - Analyse und Suche nach weiteren betroffenen Rechner im lokalen Netz
  - Meldung an Rechenzentrum / Rechtsabteilung / Provider / CERT (Koordination / Strafverfolgung)

Sicherheit - warum und wieviel?

- PRO hohe Sicherheit
  - Schutz der eigenen Ressourcen
  - Erhaltung der Verfügbarkeit
  - Schutz der Vertraulichkeit
  - Verhinderung von Mißbrauch
  - Rechtliche Situation z. Zt. Unklar
- CONTRA hohe Sicherheit
  - Fragwürdigkeit des Nutzens
  - Einschränkung des bequemen Arbeitens
  - Implementierung zeitaufwendig und schwierig
  - Widerspricht dem offenen Charakter eines Hochschulnetzes



Probleme:
- Entscheidungsfindung
- Konzept
- Realisierung
- Maintenance

Lokale Rechnersicherheit

- Physikalische Sicherheit
- Grundinstallation
- Backups: Die wichtigsten Kommandos
- Paßwortsicherheit
- Zugriffsrechte

Physikalische Sicherheit

- Zugang zum Rechner bzw. der Rechnerhardware sichern
- BIOS-Paßwort setzen
- Bootreihenfolge fest auf Festplatte einstellen
- Paßwort für Bootloader setzen (z.B. LILO, GRUB)
- Rechner vor unbeabsichtigtem Herunterfahren sichern

„Bootaccess = Rootaccess“

Grundinstallation

VOR einer Installation

- Rechner bei Grundinstallation nicht am Netz betreiben
- Patches und Updates nur aus vertrauenswürdigen Quellen beziehen
- Checksummen der Updates mit den Herstellerangaben vergleichen (z.B. md5 Summen, Signaturen)
- Security Mailinglisten des Herstellers abonnieren
- Lokale Security Mailinglisten abonnieren
- Geeignete Partitionierung mit getrenntem /, /var/log und /tmp überlegen
- Geeignete Filesysteme für die Partitionen auswählen

Grundinstallation

- Updates einspielen.
- Nach Einspielen von Updates Konfigurationsdateien überprüfen (neue Defaults möglich)
- Nichtbenötigte Dienste deaktivieren (Voranstellen eines # vor die entsprechende Zeile in /etc/inetd.conf)
- Nicht benötigte Start- / Stopscripten deaktivieren (/etc/rc.d/, /etc/init.d/)
- Tcpwrapper in /etc/inetd.conf für Zugangskontrolle und Logging konfigurieren
- /etc/hosts.equiv löschen, falls die r* Utitlities nicht verwendet werden (ssh bietet eine sichere Alternative)

Grundinstallation

- Nicht benötigte Systemaccounts / -gruppen deaktivieren (aus /etc/passwd entfernen oder beim Paßword * voranstellen)
- Hardening Scripten ablaufen lassen
- Interaktive Remotelogins für Systemaccounts abschalten (/etc/login.access oder /etc/ftpusers)
- Verbiete lpd und syslogd Verbindungen aus dem Netz anzunehmen
- /etc/hosts.allow und /etc/hosts.deny restriktiv konfigurieren
- Suchpfad von root darf das aktuelle Verzeichnis NICHT enthalten
- Virenscanner installieren!
- S.M.A.R.T. aktivieren

Grundinstallation

OpenSSH absichern:

- UsePrivilegeSeparation yes
- Protocol 2
- PasswordAuthentication no
- PermitRootLogin {without-password | forced-commands-only | no }
- Für interaktives Arbeiten „sudo“ statt Root-Zugriff per SSH?
- Für automatisierte Aktionen spezielle SSH-Keys mit Einschränkungen in ~/.ssh/authorized_keys eintragen

Grundinstallation

- Nach einem Reboot die laufenden Dienste überprüfen (netstat -an --inet, lsof -i und ps aux)
- Remote logging in /etc/syslog.conf für wichtige Logeinträge
- Sicherstellen daß alle von root bzw. von den Start-/Stopscripten verwendeten Dateien und ALLER ihrer übergeordneten Verzeichnisse als Besitzer root haben und nur root Schreibrechte besitzt
- Verwendung absoluter Dateinamen in Scripten
- Backup bzw. Image des kompletten Systems anlegen (wichtig für Wiederherstellung bzw. Überprüfen des Rechners)

Grundinstallation

- Nach erfolgreicher Grundinstallation mit tripwire oder aide Datenbank mit Informationen über den Zustand des Dateisystems erstellen
- Erstellen einer Liste aller Dateien und Verzeichnisse mit „ls -laR /“ (Zugriffszeiten, Größe und Mode) und md5-Summen (Integrität) mit „find / -type f -exec md5sum {} \;“
- Die Datenbank bzw. die Liste unbedingt auf einem schreibgeschützen Datenträger aufbewahren.
- Mit Hilfe der so gewonnen Daten lassen sich Veränderungen am Dateisystem feststellen (evtl. nur mit Hilfe eines Rescue-Systems)

Backup

- Übersicht über die wichtigsten Kommandos:

Kommando	Erklärung
tar	Archivieren und Installieren von Dateien
cpio	Kopieren von Dateien auf die Standardausgabe
dd	Low-level Kopie (Byte für Byte)
dump	Erzeugt ein Backup einer Partition
restore	Einspielen von Backups, die mit dump erzeugt wurden

- regelmäßige Backups sind essentiell (z.B. bei Sicherheitsproblemen)
- Day 0, komplett und inkrementell (z.B. mit tar)
- Vollständige Backups sollten aufgehoben werden, um frühere Systemmanipulationen nachvollziehen zu können (z.B. monatliche Backups)
- Auf wechselnde Medien schreiben!
- Wiederherstellung testen

Backup

Was man bei Backups unter Unix bedenken sollte:

- Bei einem Backup werden i.d.R. die Zeitstempel in den Inodes geändert. Nur beim Low-level Zugriff bzw. bei einem readonly gemounteten Dateisystem bleiben sie unverändert.
- Beim dateiweisen Kopieren benötigt man durch „sparse files“ auf dem Zielmedium mehr Platz (ls -l und ls -s).
- Das Verzeichnis /etc sollte immer komplett gebackuped werden.
- Die Backupmedien müssen immer sicher verwahrt bzw. vernichtet werden (Medien verschlüsseln?).

Backup

„Poor man's Backup“:

```
# cp -al <altes_dir> <neues_dir>
# rsync <zu_sicherndes_dir> <neues_dir>
```

Vorteile:

- Jedes Verzeichnis enthält ein Full Backup
- Jedes Verzeichnis belegt nur den Platz den die Änderungen gegenüber dem vorigen Snapshot belegen, da rsync automatisch die Hardlinks auflöst
- Backup per ssh/rsync über Netzwerk möglich

Nachteil:

- Einzelne Dateien nur manuell wiederherstellbar

Zugangskontrolle über das Paßwort

Probleme:

- Kenntnis eines Paßworts verschafft den Zugang zum System und definiert dort die Rechte
- Offene Accounts erlauben den Zugang ohne Paßwort (bei SGI üblich)
- Sniffer im Netz können das Paßwort mithören, falls es im Klartext übertragen wird (telnet, ftp, pop, ...)
- Ist jemand im Besitz des Paßwortfiles, so kann er versuchen mittels Crack, John the Ripper, ... weitere Paßworte zu ermitteln

Elemente des Paßwortfiles

Feld	Funktion	Beschreibung
1	Login Name	Login-Name des Benutzer, maximal 8 Zeichen
2	Password	Verschlüsseltes Paßwort.
3	User ID	Benutzernummer zur Identifikation im System (UID). Anzeige mit **id.**
4	Group ID	Gruppennummer des Benutzers (GID) gestattet Gruppenzugriff auf Files.
5	Personal	Freitext Beschreibung des Benutzers.
6	Login directory	Pfad des Login-Verzeichnisses des Benutzers
7	Login shell	Login-Shell des Benutzers, kann mit **chsh** vom Benutzer geändert werden.

Wichtig:

- keine offenen Accounts
- UID eindeutig vergeben
- UID=0 gibt Rootrechte

Empfehlung:

- regelmäßiger Konsistenz-check mit z.B. TARA

 # cat /etc/passwd

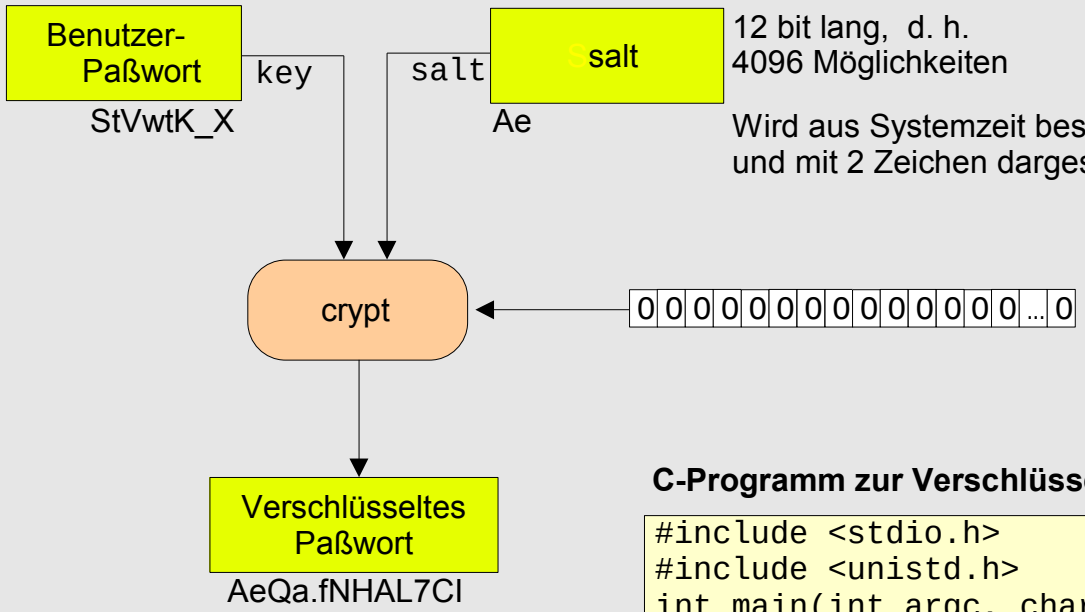
root:gw1oNN0pIlsFE:0:0:Superuser Account:/root:/bin/bash

bin:x:1:1:bin:/bin:/bin/bash

daemon:*:2:2:Daemon:/sbin:/bin/bash

Wie entsteht das verschlüsselte Paßwort

Mechanismus



C-Programm zur Verschlüsselung: `# a.out key salt`

```
#include <stdio.h>
#include <unistd.h>
int main(int argc, char *argv[])
{
    /*                          key     salt     */
    printf(„\n %s \n\n“,crypt(argv[1],argv[2]) );
    return 0;
}
```

Gefahren durch Crack

- Verschlüsselt Worte und Kombinaten von Worten und vergleicht die Ergebnisse mit den Einträgen in /etc/passwd.
- Crack verwendet Wörterbücher und einfache, aber bei Benutzern übliche Variationen wie:
  - i,l,t nach 1
  - E nach 3
  - o nach 0 (NULL)
- Weiterhin wird das Anhängen von Ziffern und Spiegeln erkannt.
- Für ein Paßwort mit 6 Großbuchstaben benötigt Crack weniger als 4 Stunden bei einem Brute-Force Angriff
- John the Ripper ist wesentlich leistungsfähiger!

Schutz des Paßwortfiles

Tips zum Paßwortfile:

- Regelmäßig Paßworte mit Crack testen
- Password-aging einsetzen?
- Benutzer zwingen Sonderzeichen zu verwenden
- Zugriff auf /etc/passwd für tftp und ftp verbieten
- Einsatz von Shadow Password Files da lokale Benutzer das Paßwortfile lesen können
- suchen nach inkonsistenten (UID=0) oder offenen Accounts

```
# awk  -F: '{ if ($2==““) print $1 }' /etc/passwd
# awk  -F: '{ if ($3==“0“) print $1 }' /etc/passwd
```

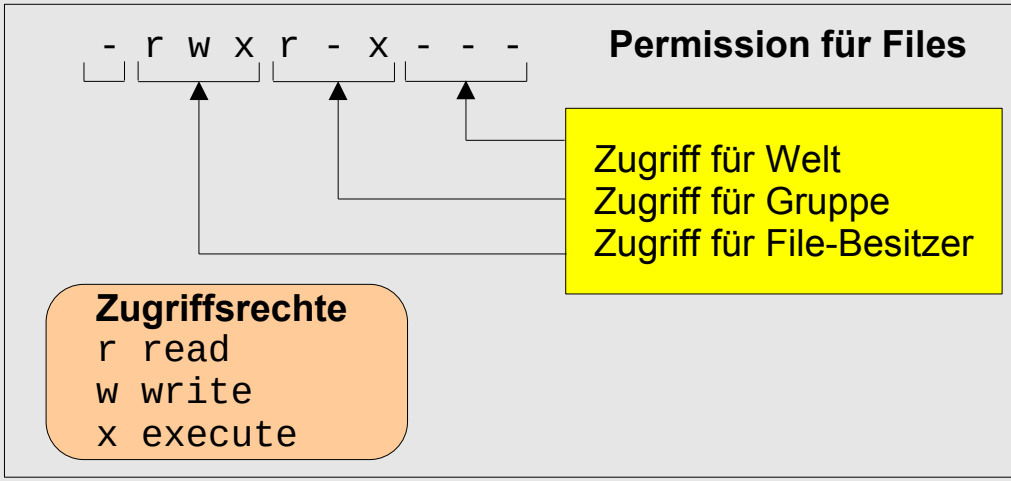
Zugriffsrechte

Permission bits für chmod

octal	binary	perms
0	000	- - -
1	001	- - x
2	010	- w -
3	011	- w x
4	100	r - -
5	101	r - x
6	110	r w -
7	111	r w x

Zuordnung für umask

octal	binary	perms
0	000	r w x
1	001	r w -
2	010	r - x
3	011	r - -
4	100	- w x
5	101	- w -
6	110	- - x
7	111	- - -

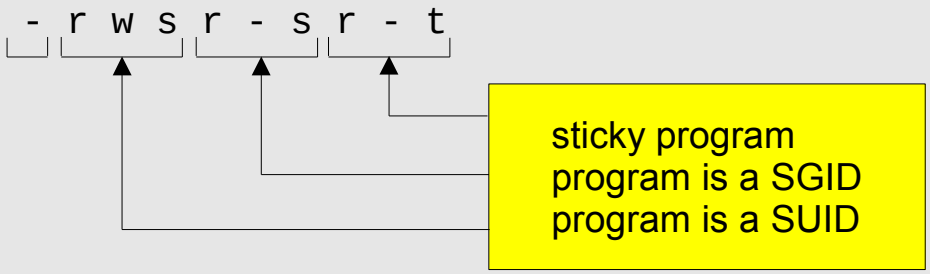


Permissions für Verzeichnisse
--x = search bit: Erlaubt das Betreten, aber kein ls
r-x = erlaubt das Betreten und Anzeigen

Veränderung der Zugriffsrechte mit chmod

```
# chmod [-fR] absolute_mode file
# chmod [-fR] [who]+permission...   file...
# chmod [-fR] [who]-permission...   file...
# chmod [-fR] [who]=[permission...] file...
```

Spezielle Permissions: SUID root



Das SUID Bit
s = SUID (octal = 4000)
s = SGID (octal = 2000)
t = sticky bit (octal = 1000)

Welche SUID-Programme sind gefährlich?

- im Prinzip alle
- insbesondere Programme wie sendmail, wo es immer wieder neue Lücken gibt

Beispiel eines einfachen Mißbrauchs

- Eingeloggter Superuser läßt sein Terminal kurz unbeaufsichtigt:

```
#root> cp /bin/ksh /tmp/.my-su-sh
#root> chmod 4755 /tmp/.my-su-sh
```

- d. h. 2 Befehle genügen, um sich längerfristig root-Privilegien zu verschaffen

Spezielle Permissions: SUID root

Funktionalität und Gefährlichkeit

- Programme mit SUID-root laufen beim Ablauf mit den Provilegien von root, anders als „normale“ Programme, die unter der UID dessen laufen, der das Programm aufruft.
- Notwendig ist der SUID-Mechanismus für viele Programme, z. B. das Programm passwd (Ändern des Paßworts).
- Die Gefahr besteht in der „unsauberen“ Ausführung der Programme, die dann den root-Zugang eröffnen (z. B. Buffer overflow, IFS-Bug).
- Selbst nach vielen Jahren werden oft noch Schwachstellen in SUID-Programmen gefunden.
- Shell-Skripten dürfen auf keinen Fall mit SUID-root versehen werden. (Shell-Escape!)

SUID-/GUID-Programme unter SuSE Linux 7.3

/bin/su
/bin/ping
/bin/eject
/bin/mount
/bin/ping6
/bin/umount
/opt/kde2/bin/konsole
/opt/kde2/bin/artswrapper
/opt/kde2/bin/kdesud
/opt/kde2/bin/kreatecd
/opt/kde2/bin/kcheckpass
/opt/kde2/bin/konsole_grantpty
/usr/bin/at
/usr/bin/rcp
/usr/bin/rsh
/usr/bin/ssh
/usr/bin/bing
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/wall
/usr/bin/lppasswd
/usr/bin/crontab
/usr/bin/chage
/usr/bin/mandb
/usr/bin/write
/usr/bin/ziptool
/usr/bin/expiry
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/rlogin
/usr/bin/screen
/usr/lib/mc/bin/cons.saver
/usr/lib/pt_chown
/usr/sbin/pppd
/usr/sbin/traceroute6
/usr/sbin/traceroute
/usr/sbin/sendmail
/usr/sbin/suexec
/usr/X11R6/bin/dga
/usr/X11R6/bin/Xwrapper
/usr/X11R6/bin/xlock-mesa
/usr/X11R6/bin/xlock
/usr/X11R6/bin/xterm
/usr/X11R6/bin/v4l-conf
/usr/X11R6/bin/XFree86
/sbin/unix_chkpwd

Suche nach Files und Verzeichnissen

- Welt-/ Gruppenschreibbare Verzeichnisse sind nur für /tmp, /usr/tmp etc. sinnvoll und notwendig.

```
# find / -perm -2 \! -type d -print     für Welt schreibbar
# find / -perm -20 \! -type d -print    für Gruppe schreibbar
```

- SUID / SGID-Berechtigungen sollten nur bei wohldefinierten Programmen vorhanden sein.

```
# find / \( -perm -4000 -o -perm -2000 \) -type f -print
```

- Sinnvollerweise generiert man bei jedem Suchlauf Listen und vergleicht diese mit diff.
- externe Filesysteme sollten mit -nosuid oder gleich mit -noexec gemountet werden.

Permissions von Special Files

- Device-Files bilden die Schnittstelle des Kernels zu den einzelnen Hardware-Devices.
- Lese- und Schreibrechte auf ein Special File bedeuten entsprechende Rechte auf das Device
- Beispielsweise können über /dev/kmem Systemaktivitäten abgehört und modifiziert werden oder über /dev/hda low-level auf die Festplatte zugegriffen werden.
- Special Files sind auch außerhalb von /dev gültig!

```
# find / \( -type c -o -type b \) -exec ls -l {} \;
```

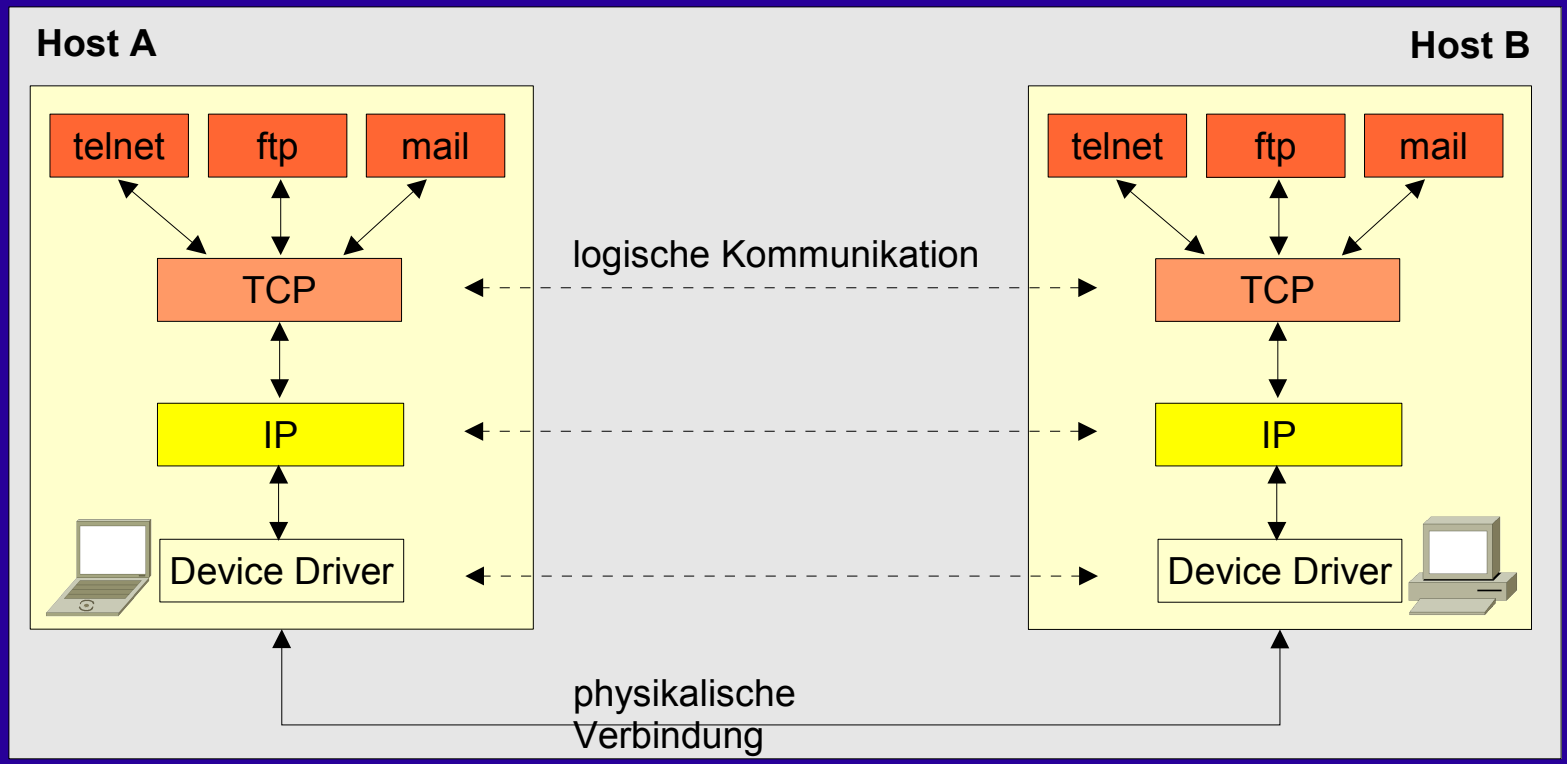
crontab und at

- crontab und at ermöglichen die periodische bzw. einmalige Ausführung von Kommandos zu bestimmten Zeiten
- Bei falschen Berechtigungen auf /var/spool/cron/tabs oder /var/spool/atjobs können lokale Benutzer dort root ein Script unterschieben
- Evtl. at deaktivieren und crontab über crontab.allow bzw. crontab.deny absichern

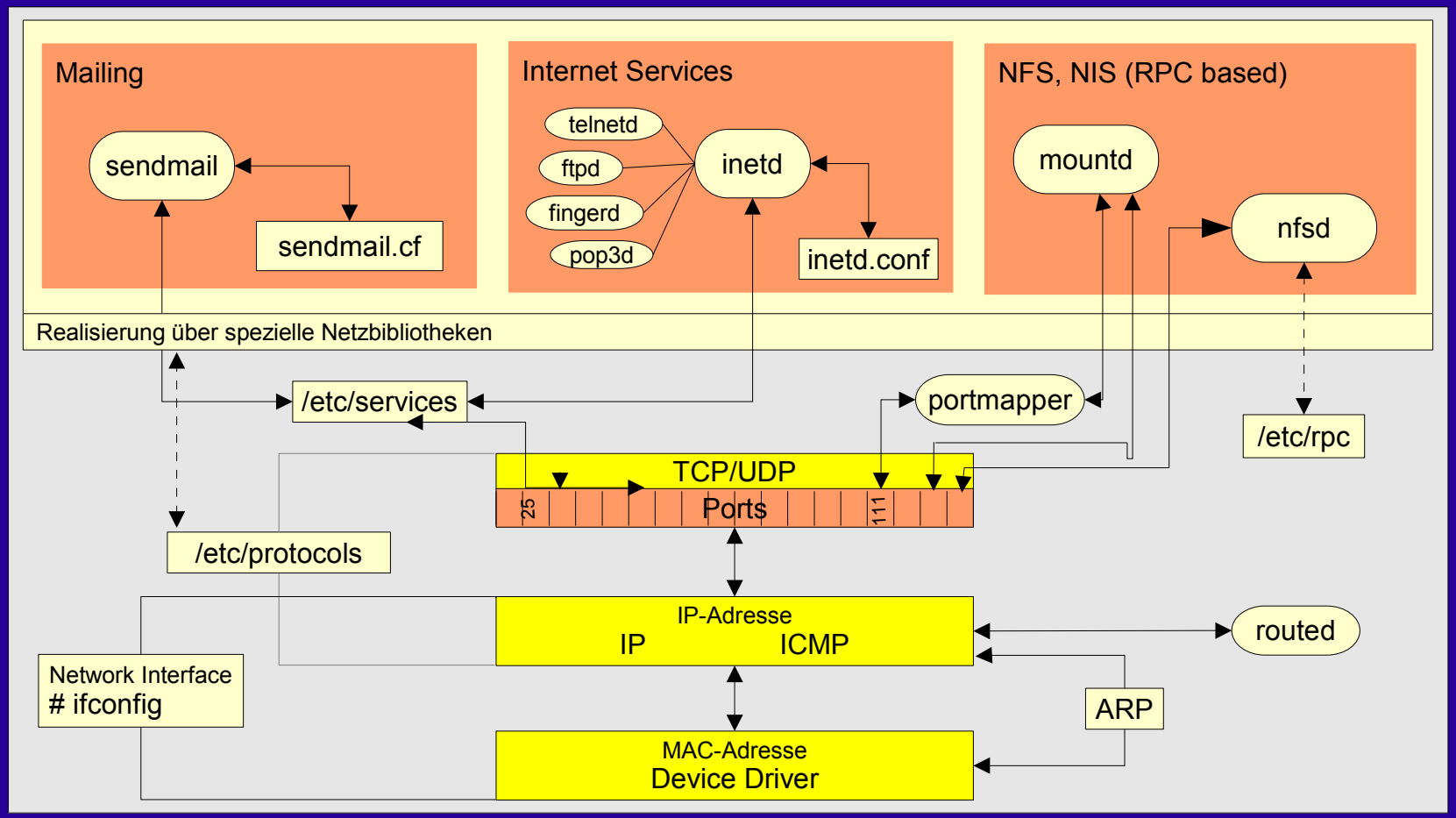
Netzwerksicherheit eines Rechners

- TCP-IP
- Gefahren aus dem Netz
- Anfällige Dienste
- geschwätzige Dienste
- Tcp-Wrapper und PD-Portmapper
- Personal Firewall

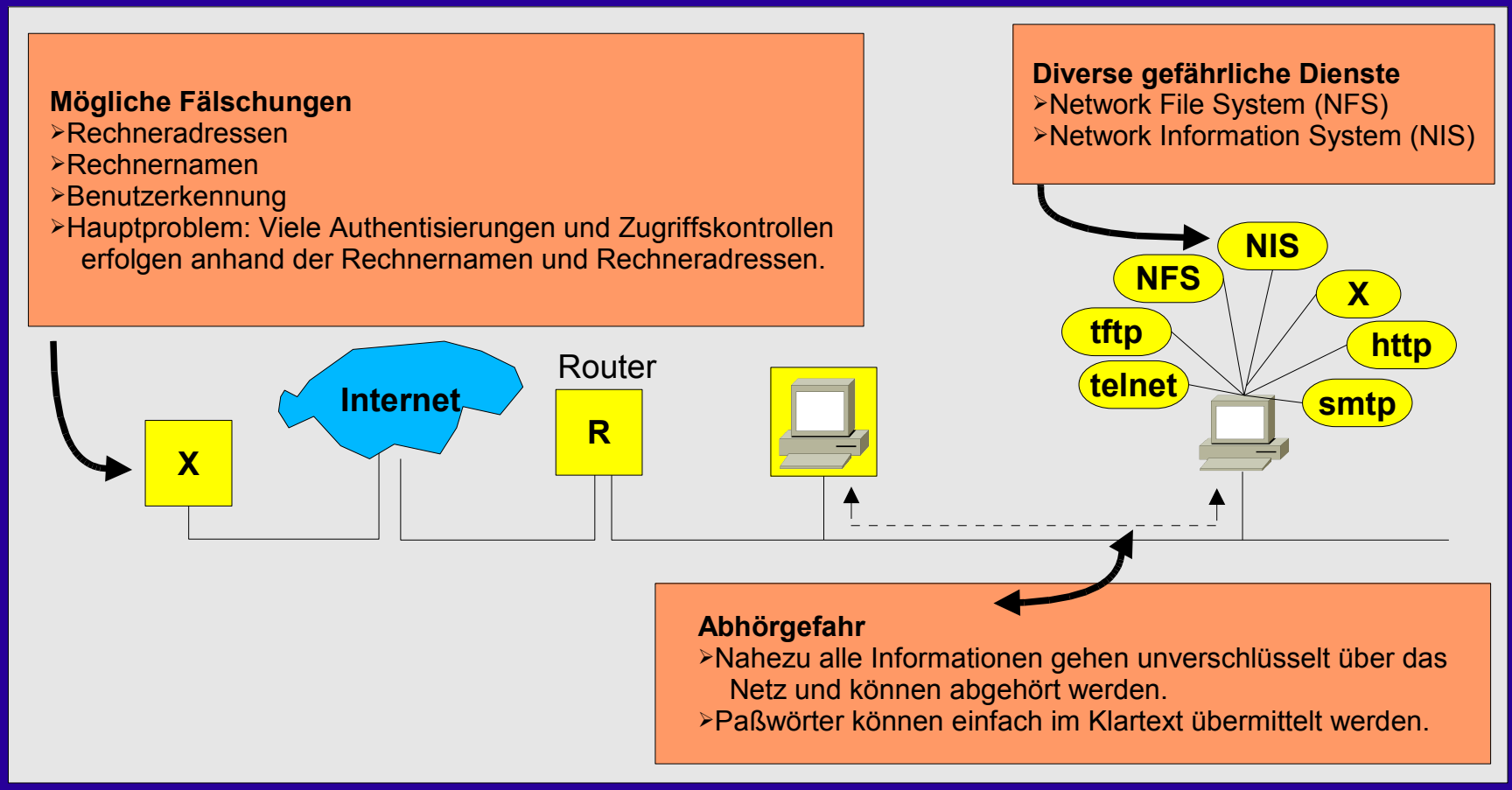
TCP-IP Protokoll



Unix-Implementierung der Netzwerkdienste



Übersicht: Gefahren aus dem Netz



Denial of Service

- der Rechner wird vorübergehend in seiner Funktionstüchtigkeit stark beeinträchtigt
- es kann zu einem Absturz kommen
- es werden (i.d.R.) keine Rechte auf dem Rechner erlangt
- Gegenmaßnahmen
  - Patches einspielen
  - Pakete am Router stoppen
- Typische Mechanismen
  - Überlastung des Rechners durch Fluten mit SYNs oder UDP-Paketen
  - Ausnutzung von Implementierungsfehlern (speziellen Paketen, Fragmentierung)

Sniffer

- Viele Informationen gehen unverschlüsselt übers Netz und können leicht abgehört werden (z.B. Benutzername und Paßwort)
- Durch die abgehörten Login-Informationen hat man Zugriff auf die entsprechenden Ressourcen
- Gegenmaßnahmen:
  - Paßwörter nur über „sichere“ Netze verwenden
  - Versuchen das lokale Netz gegen Sniffer zu sichern
  - Einsatz z.B. von Secure Shell, https, imaps etc.
- Sniffer sind im Netz kaum zu erkennen (da passiver Angriff)
- Sniffer können auch über einen Switch hinweg mithören
- Es gibt oft keine Alternativen zu Klartext-Paßwort

IP-Spoofing

- Fälschung von IP-Adressen über Router-Grenzen hinweg
- Ausnutzung von Diensten über eine gefälschte IP-Authentisierung
- Oft DOS-Attacke auf vorgegebenen Host nötig
- Source-Routed Frames ermöglichen Angriffe aus dem Internet
- Verwundbar: alle Dienste mit Authentifizierung über IP

  (r-Dienste, NFS, TCP-Wrapper, X, ...)
- Ermöglicht Connection Hijacking und Command Insertion
- Gegenmaßnahme: Routerkonfiguration um Source-Routed-Frames und gespoofte IP-Adressen abzuweisen

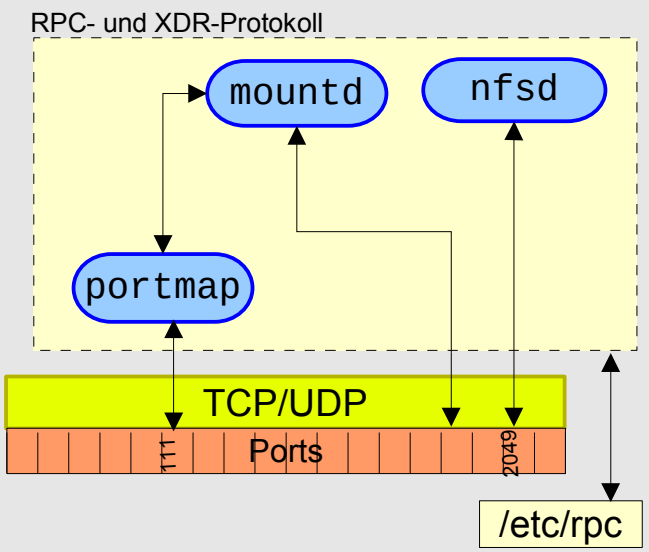
Angriff über Bande

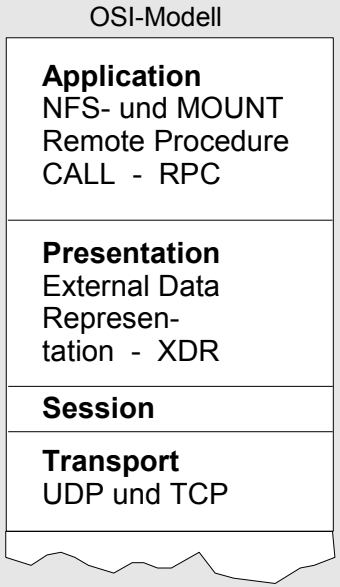
- DNS-Cache Poisoning:
  - Füttern von Nameservern mit falschen Informationen
- direkte Manipulation von Nameservern
- Man-in-the-Middle Attacken
  - bei Serverdiensten
  - Routing
- Gegenmaßnahmen:
  - Verwendung von Kryptographisch abgesicherten Protokollen mit bekannten Zertifikaten der Gegenstelle
  - Eintragen wichtiger IP-Adressen in eigene /etc/hosts Datei
  - Statische ARP Einträge (/etc/ethers)

Absicherung des X-Servers

- Möglichkeiten und Probleme
  - Der X-Server kontrolliert Bildschirm, Maus und Tastatur
  - Ist der X-Server ungesichert übers Netz erreichbar, können alle Events mitprotokolliert werden bzw. externe Eingaben eingeschleußt werden
- Absicherung:
  - Hostbezogen: xhost [+|-] [hostname] — Anfällig für IP-Spoofing
  - Benutzerbezogen: xauth magic cookies — Cookie kann abgehört werden
  - Einsatz von SSH-Tunneln

RPC-basierende Dienste und portmapper





portmapper Funktionen

1. **SET:** Registrieren eines Dienstes
2. **UNSET:** Entfernen eines Dienstes aus der Tabelle
3. **GETPORT:** Abfrage der Portnummer eines Dienstes
4. **DUMP:** Zugriff auf den gesamten Inhalt der Portmapper-Tabelle

Abfragen der portmapper-Tabelle (mittels DUMP bzw. rpcinfo)

```
# rpcinfo -p [targethostname]
program vers proto port

  100000  2  tcp   111  portmapper
  100000  2  udp   111  portmapper
  100005  1  udp  1029  mountd
  100005  3  udp  1029  mountd
  100003  2  udp  2049  nfs
  100003  3  udp  2049  nfs
  100005  1  tcp   977  mountd
  100005  3  tcp   977  mountd
  100004  2  udp   974  ypserv
  100004  2  tcp   975  ypserv
  100007  2  tcp  2978  ypbind
  100007  2  udp  4718  ypbind
  100009  1  udp  1023  yppasswdd
  100069  1  udp   789  ypxfrd
  100069  1  tcp   791  ypxfrd
```

Sicherheitsprobleme des portmappers

- Mittels rpcinfo können die RPC-Portnummern (weltweit) abgefragt werden
- Mit UNSET kann versucht werden, Dienste aus der Registrierungs-tabelle zu entfernen (d.h. DOS Angriff)
- Die meisten Portmapper ermöglichen die lokale Weiterleitung einer Anfrage. Diese werden dann wie lokale Anfragen (mit entsprechen-den Rechten) behandelt
- Verzicht auf portmapper wegen z.B. NFS und NIS kaum möglich
- Abhilfe durch Einsatz eines möglichst sicheren Portmappers
- Am Router den Port 111 sperren

Sicherheitsprobleme von NIS

- Ermöglicht zentrale Verwaltung eines Clusters
- Ist konzeptionell unsicher:
  - arbeitet auf dynamischen Ports und ist deshalb schwer zu filtern
  - Zugriffskontrolle nur über NIS-Domainname
  - die Kenntnis des NIS-Domainnamens erlaubt Zugriff auf alle NIS-Daten, d.h. insbesondere auf das Paßwortfile
  - die Kenntnis des NIS-Domainnamens erlaubt fremden Rechnern sich in den Verbund einzuhängen
- Wichtig: NIS-Domainname sollte nichts mit Netzdomainnamen gemeinsam haben

Schutz von NFS

- Ermöglicht zentrale Datenhaltung
- /etc/exports verwendet IP/Hostname basierte Authentifizierung
- Zugriff auf Mountpoint über leicht zu erratendes statisches File-handle (/etc/exports ist bei Kenntnis des Filehandles wirkungslos)
- Vorsichtsmaßnahmen:
  - mit fsirand bessere Filehandles generieren
  - nur wenige Filessysteme (möglichst) readonly exportieren
  - Server darf keinen Rootzugriff durch den Client erlauben (root_sqash)
  - der Client sollte das Filesystem mit -o nosuid,nodev mounten
  - Sperren des Ports 2049 am Router
  - Einschränkung des Portmapper-Zugriffs
  - Keine Freigabe für localhost

Geschwätzige Dienste

- diverse Dienste liefern Informationen über System, Benutzer, Prozesse und Netzverbindungen
- Diese für den lokalen Gebrauch wichtigen Informationen erleichtern einen Angriff ungemein.
- Dienste die über den inetd gestartet werden:

  netstat, finger, systat (telnet <host> <dienst>)
- Weitere mögliche „Lecks“:
  - showmount -e host — Gibt Informationen über exportierte Filesysteme
  - rpcinfo - host — Liefert Liste der angebotenen RPC-Dienste
  - Banner von Diensten (WWW, Telnet, Sendmail, ...)

TCP-Wrapper und PD-Portmapper

- TCP-Wraper
  - wird in der /etc/inetd.conf dem inetd-Daemon vorgeschaltet
  - bietet eine erweiterte Logging Funktionalität
  - Bietet eine IP-basierte Zugriffskontrolle
- PD-Portmapper
  - beseitigt insbesondere die Weiterleitung von Anfragen

- Beide verwenden /etc/hosts.allow und /etc/hosts.deny zur Zugriffskontrolle
- TCP-Wrapper Support kann in viele Applikationen (z.B. SSH) eingebaut werden

Personal Firewall

- Kann einen vorgeschaltenen Firewall ergänzen
- Kann dienstunabhängig lokal die gefährlichsten Ports sperren
- Kann Zugriffe auf Ports über syslog protokollieren
- für Linux, BSD und Solaris verfügbar
- Hauptprobleme:
  - Aufstellen geeigneter Regeln
  - Überprüfen der Regeln
  - hilft wenig gegen Spoofing

iptables

- Paketfilter mit „Statefull Inspection“, „Protocol Helpern“, NAT und PAT (Port Address Translation)
- Standardmässig bereits im Kernel enthalten
- Arbeitet auf Layer 3 und 4
- Kernelspace: netfilter Kernelmodule klinken sich in Paketverarbeitung
- Userspace: iptables als Nachfolger von ipfwadm, ipchains zur Steuerung von Netfilter
- Durch modularen Aufbau funktional vielseitig erweiterbar
- www.netfilter.org
- http://www.aptalaska.net/~jclive/IPTablesFlowChart.pdf

iptables

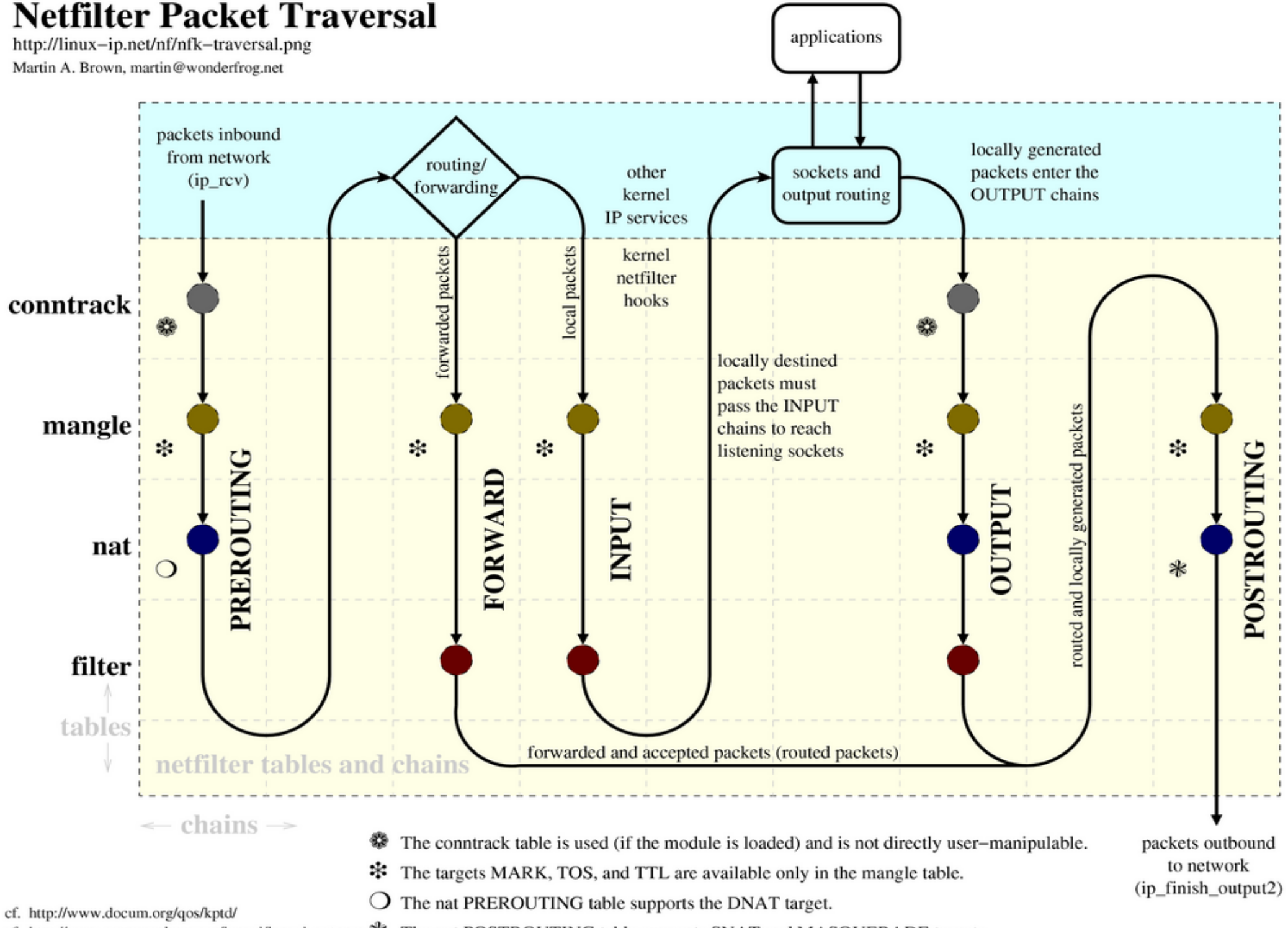
Aufbau

- Tablellen (Tables): filter (default), nat, mangle (conntrack)
- Regelketten (Chains):
  - Built-in Chains PREROUTING, INPUT, OUTPUT, FORWARD, POSTROUTING mit „Default Policy“
  - Eigene Ketten
- Regeln bestehen aus:
  - Beim Anlegen Bezug zu Ketten/Tabellen
  - Bedingungen (Match), die ein Paket erfüllen muss
  - Ziel (Target)

Netfilter Packet Traversal

http://linux-ip.net/nf/nfk-traversal.png

Martin A. Brown, martin@wonderfrog.net



❁ The conntrack table is used (if the module is loaded) and is not directly user-manipulable.

✻ The targets MARK, TOS, and TTL are available only in the mangle table.

❍ The nat PREROUTING table supports the DNAT target.

✳ The nat POSTROUTING table supports SNAT and MASQUERADE targets.

cf. http://www.docum.org/qos/kptd/

cf. http://open-source.arkoon.net/kernel/kernel_net.png

cf. http://iptables-tutorial.frozentux.net/

iptables

Filterung (Match) möglich nach:

- MAC, IP/Netz, Port jeweils Ziel und/oder Quelle
- State (NEW, RELATED, ESTABLISHED, INVALID)
- Protokoll (tcp, udp, icmp, ...)
- Ein/Ausgehendes Interface
- Paketinhalt, Rate, TCP-Flags

Erweiterungen der Netfiltermodule im Kernel ermöglichen eine Vielfalt an weiteren Matches.

iptables

Mögliche Ziele (Targets)

- Standard-Targets: ACCEPT, DROP, QUEUE, RETURN
- Erweiterungen: REJECT, LOG, DNAT, SNAT, MASQUERADE
- Eigene Ketten anspringen. RETURN verlässt eine Unterkette.
- Pakete verändern / markieren mittels „mangle“
- Transparenter Proxy mittels REDIRECT

iptables

Wichtige Eigenheiten:

- Loopback Interface muss für systeminterne Kommunikation freigegeben sein!

```
# iptables -I INPUT -i lo -j ACCEPT
```

- Bei SNAT/DNAT werden alle Pakete einer Verbindung im conntrack anhand des ersten Pakets klassifiziert!
- Die Zustände NEW, RELATED, ESTABLISHED beziehen sich auf conntrack und nicht auf TCP-FLAGS!
- Regeln werden sequentiell abgearbeitet bis ein DROP, ACCEPT oder REJECT getroffen wird.
- Trifft in den Standardketten keine finale Regel zu, dann entscheidet die Default Policy!

iptables

- Regeln ändern, einfügen, löschen

  # iptables [-t <table>] {-R | -A | -I | -D } <chain> [<match>] -j <target>
- Regelketten anzeigen, leeren, Zähler zurücksetzen

  # iptables [-t <table>] {-L | -F | -Z } [chain]
- Eigene Regelketten erstellen/löschen

  # iptables [-t <table>] {-N | -X } <chain name>
- Default Policy setzen

  # iptables [-t <table>] -P <chain> <ACCEPT | DROP | REJECT>
- Kompletten Regelsatz sichern, laden

  # iptables-save > datei

  # iptables-restore < datei
- Beispielregel

  # iptables -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu

Einschränkungen für ROOT

- Problem:
  - root-account unter Unix hat uneingeschränkten Zugriff auf das System
  - ein Angreifer mit Rootberechtigung kann das System verändern (Log-/Konfigurationsdateien ändern/löschen, Prozesse starten, Kernel Module laden, ...)
  - Wie kann man ein System vor noch unbekannten Angriffen schützen?
- Lösung:
  - Verwenden von „chattr“ im ext2 und ext3 Dateisystemen
  - Linux Intrusion Detection System (LIDS) Kernel Patch
  - Saint Jude / Saint Michael
  - Security-Enhanced Linux SELinux Kernel Module
  - Ausführbaren Code in Stackbereichen verbieten mit W^X, Exec Shield, ...

Einschränkungen für ROOT

- Linux Intrusion Detection System (LIDS) Kernel Patch
- Ermöglicht:
  - Versiegeln des Kernels
  - Kernelbasierte ACLs für Dateien und Prozesse
  - Kernelbasiertes Erkennen von Portscans
  - Schutzmechanismen werden beim Booten aktiviert und gelten auch für root
- Problem:
  - Schwierig einzustellen

Einschränkungen für ROOT

- Saint Jude / Saint Michael sollen ein System gegen unbekannte Gefahren schützen
- Saint Jude
  - Kernel Modul für Linux (2.2, 2.4, 2.6?) und Solaris 8 Sparc (32/64 Bit)
  - Erkennt und verhindert Root-Exploits nach Trainingsphase durch das Beobachten von Privilegänderungen
- Saint Michael
  - Kernel Modul für Linux 2.4, 2.6?
  - soll Kernel Module Rootkits erkennen
  - soll Kernelstrukturen wiederherstellen

Erkennen von Vorfällen

- Größtes Problem: Die unbemerkte Modifikation des Systems
- Möglichkeiten Angriffe (und Modifikationen) zu erkennen:
  - lokales und zentrales Logging mit automatischer Auswertung und Benachrichtigung
  - regelmäßige Überprüfung der Filesystemintegrität (rpm -V, md5sum, tripwire)
  - lokale und remote Überprüfung der Rechnerkonfiguration (netstat, Portscans)
  - Analyse des Netzwerkdatenverkehrs
  - Host- und Netzwerkbasierte Intrusion Detection Systeme
  - Beobachten von Ports auf Portscans
  - Überprüfen von Programmen mit „strings“ oder „strace“
  - chkrootkit

Analyse der Logdateien

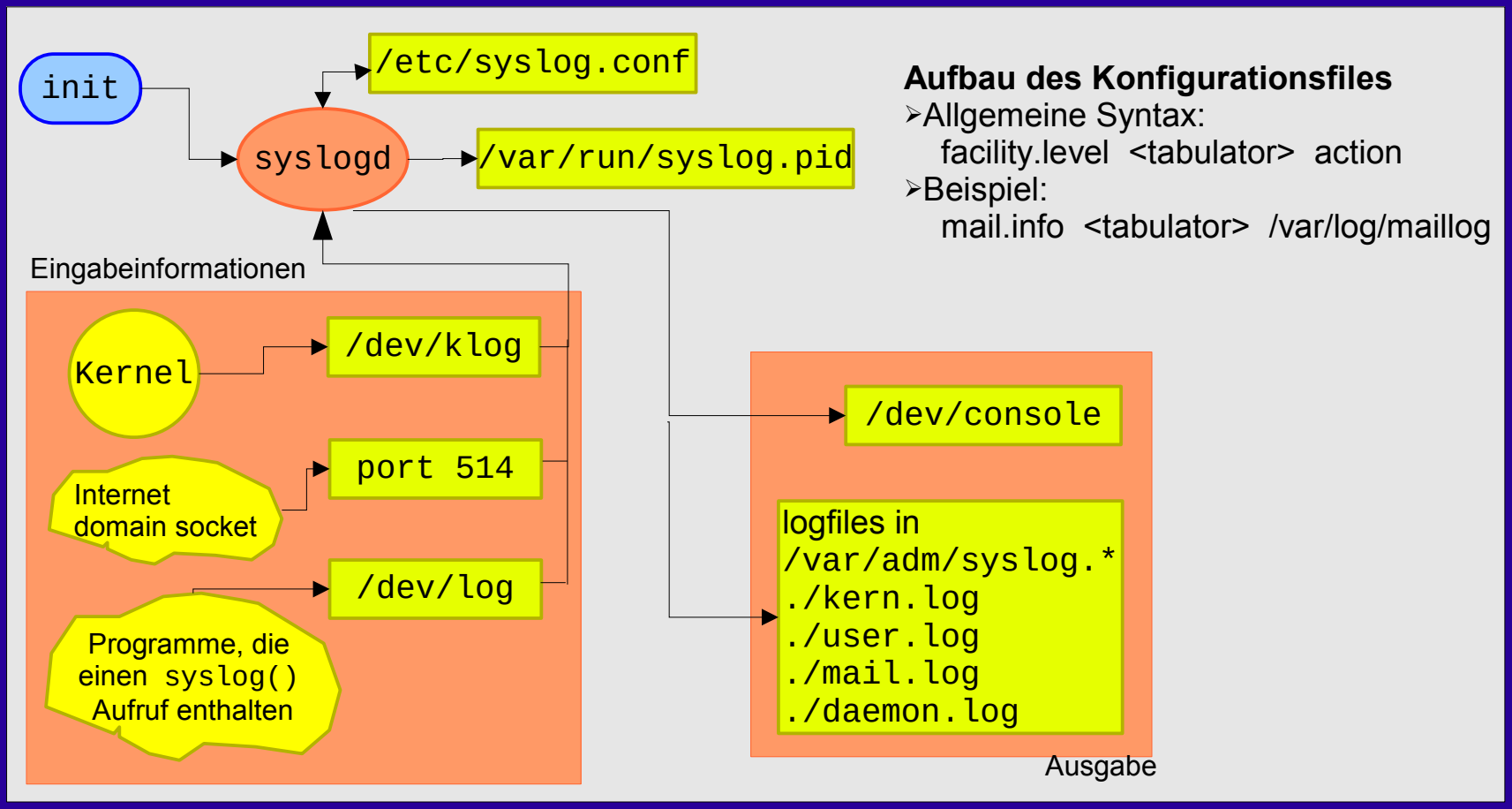
- Diverse Logdateien
  - utmp
  - wtmp
  - lastlog
  - acct
  - sulog
- Protokollierung über syslog / syslog-ng

Problem:

- Von Hand zu mühsam
- Wonach soll gesucht werden?

Lösung: Swatch oder Logsurfer

Syslog-Mechanismus



Aufbau des Konfigurationsfiles

- Allgemeine Syntax:
  facility.level <tabulator> action
- Beispiel:
  mail.info <tabulator> /var/log/maillog

Syslog-Konfiguration

Facility	Erklärung
kern	generated by the kernel
user	generated by user processes (default facility)
mail	generated by the mail system
daemon	generated by system daemons
auth	generated by the authoritation system: login, su, etc.
lpr	generated by the printer spooling system
local0 ... local7	reserved for local use

priority	Erklärung
LOG_EMERG	a panic condition reported to all users
LOG_ALERT	a condition to be corrected immediately
LOG_CRIT	critical condition
LOG_ERR	errors
LOG_WARNING	warning messages
LOG_NOTICE	condition requiring special handling
LOG_INFO	general information messages
LOG_DEBUG	containing information useful for debgging

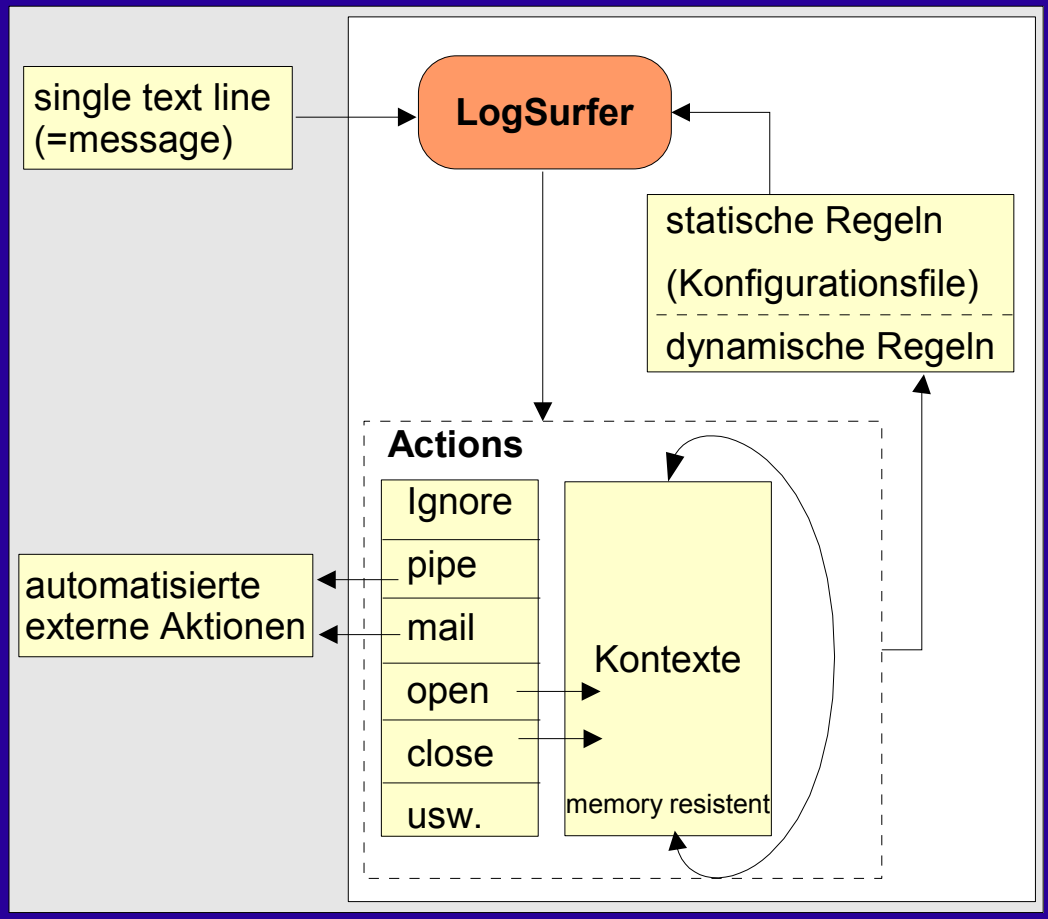
Syntax	Erklärung
/filename -/filename	Die von Syslog erzeugten Einträge werden in die angegebene Datei geschrieben (-/ unterläßt sync)
@hostname @ip-adresse	syslogd leitet die Einträge an die angegebenen Rechner weiter
user1,user2	Die Meldungen werden an die angegebenen Benutzer ausgegeben, falls diese eingeloggt sind
*	Die Meldung geht an alle eingeloggten Benutzer
|named pipe	Die Meldung wird an die named Pipe (mkfifo) gegeben

Zentrale Logdateien

- Vorteile
  zentrale Datenhaltung und Auswertung
- Nachteile
  alle Informationen gehen über das Netz.
  Der Logrechner sollte gesichert werden.
  Anzahl der Logeinträge.

Auswertung mit LogSurfer

- Grundprinzip
  - LogSurfer vergleicht Textzeilen (aus den logfiles) mit Regeln.
  - Die Regeln sind aus „regular expressions“ aufgebaut.
  - Das System veranlaßt anhand des Regelsatzes diverse Aktivitäten.
  - Im Gegensatz zu Swatch (oder grep) arbeitet LogSurfer kontextabhängig.
  - Dynamisch erzeugte Regeln möglich.



tripwire

Ziel: Festhalten aller Informationen über das Filesystem und Erkennung von Veränderungen

Funktionsweise:

- tripwire generiert nach den Vorgaben einer Konfigurationsdatei eine Datenbank mit Zugriffszeiten, Inode, Größe, Besitzer und mehreren kryptogrphischen Checksummen einer jeden Datei
- Über die Konfigurationsdatei können die zu sammelnden Informationen für jede Datei bzw. Verzeichnis einzeln festgelegt werden.
- Nach Erstellung der Datenbank kann tripwire die Differenzen zum aktuellen Status ausgeben

tripwire

- Zur Sicherheit muß tripwire, die Datenbank und die zugrunde liegenden Konfigurationsdatei auf einem „sicheren“ Medium gespeichert werden.
- Tripwire sollte statisch gelinkt sein
- Nachteil: Konfiguration ist aufwendig da sich viele Dateien im normalen Betrieb ändern
- Open Source Alternative: AIDE

lokaler Konfigurationscheck

- Rechnerkonfiguration ist komplex und zeitaufwändig
- Es können Flüchtigkeitsfehler unterlaufen
- „korrekte“ Konfiguration ist nicht trivial erreichbar
- Abhilfe durch Tools wie TARA, die scriptgesteuert das System auf bekannte Schwachstellen untersuchen und einen Report abliefern
- TARA ist einfach zu Bedienen und liefert gute Erklärungen und Vorschläge zu gefundenen Problemen
- LSAT Linux Security Auditing Tool
- Härten des Systems mit Bastille Linux

remote Überprüfung

- Gründe für eine Remote Überprüfung:
  - Gezielte scriptgesteuerte Suche nach Schwächen von Netzwerkdiensten
  - System kann manipuliert sein und zeigt mit Bordmitteln nicht mehr alle offenen Ports an
- Tools:
  - SAINT
  - SARA
  - Nessus
  - nmap

Analyse des Netzverkehrs

- Bei Netzwerkproblemen oder der Analyse von verdächtigem Datenverkehr
- tcpdump
- ethereal

Bearbeiten eines Einbruchs

- Systemintegrität unklar
- Rootkit verhindert Analyse mit Bordmitteln (versteckte Verzeichnisse, Dateien, Prozesse oder Ports)
- Auffälliger sind meist Symptome wie hohe Last, volles Dateisystem oder eine hohe Netzlast)
- Es gibt kein „Rezept“ für weitere Schritte

Bearbeiten eines Einbruchs

- Allgemeine Hinweise:
  - Logbuch zur späteren Nachvollziehbarkeit von Aktionen führen. Bei Arbeiten am Rechner den Befehl **script** verwenden.
  - Schritte welche Daten verändern nur nach guter Überlegung angehen (der erste Bearbeiter hat die größten Chancen Beweise zu sichern – oder zu vernichten).
  - Ein dediziertes Analysesystem (evtl. Laptop) ist hilfreich.
- Strategische Ziele:
  - Alle betroffenen Rechner ausfindig machen
  - Den Angriff nachvollziehbar machen
  - Benachrichtigung an weitere Betroffene
  - falls rechtliche Schritte angedacht sind eine korrekte und komplette Datensicherung

Arbeiten vor einem Shutdown

- Potentieller Selbstzerstörungsmechanismus in Shutdown integriert.
- VOR Shutdown möglichst viele Informationen aus dem laufendem System sammeln:
  - last / w
  - ps auxwww bzw. ps elfwww
  - lsof
  - ltrace, strace auf verdächtige Programme
  - find und ls -lat (ändern Timestamps!)
  - netstat -a
- Potentiell können die genannten Tools und auch Shared Libraries oder Kernelmodule durch einen Rootkit erstzt sein

Analyse des gehackten Systems

- wichtig: lowlevel Kopie mit z.B. dd anfertigen und nur mit der Kopie arbeiten. Orginal sicherstellen.
- Medium immer readonly und -o nosuid,nodev mounten
- Logbuch führen
- Verdächtige Programme mit „strings“ bearbeiten
- Verdächtige Programme nur in kontrollierter Umgebung ausführen!
- Alles weitere hängt vom speziellen Fall ab

Informationsquellen

- www.cert.org:
  - Statistiken:

    http://www.cert.org/stats/cert_stats.html
  - Unix Security Checklist:

    http://www.cert.org/tech_tips/AUSCERT_checklist2.0.html
  - Steps for Recovering from an Unix or NT Compromise:

    http://www.cert.org/tech_tips/win-UNIX-system_compromise.html
  - Unix Configuration Guidelines:

    http://www.cert.org/tech_tips/unix_configuration_guidelines.html
  - Aktuelle Angriffe auf:
    - Scans nach verwundbaren Webapplikationen
    - SSH-Scans mit Brute force

Informationsquellen

- www.sans.org:
  - Information Security Reading Room

    http://www.sans.org/rr
  - The Twenty Most Critical Internet Security Vulnerabilities

    http://www.sans.org/top20/
- NIST: Practices & Checklists, Implementation Guides
  - http://csrc.nist.gov/pcig/cig.html
- Basic Steps in Forensic Analysis of Unix Systems:

  http://staff.washington.edu/dittrich/misc/forensics/
- Das Honeynet Projekt

  http://www.honeynet.org
- http://packetstormsecurity.nl/

Informationsquellen

- www.securityfocus.com (Bugtraq)
  - Incident Response Tools For Unix, Part One: System Tools (http://www.securityfocus.com/infocus/1679)
  - Incident Response Tools For Unix, Part Two: File-System Tools (http://www.securityfocus.com/infocus/1738)
- Common Vulnerabilites and Exposures
  - http://cve.mitre.org/
- http://www.first.org/resources/guides/reference.html
- http://www.frsirt.com/exploits/
- http://www.phrack.org/ (Hacker magazine)
- http://www.freshmeat.net (Software)

Informationsquellen

- Practical UNIX Security, Simson Garfinkel and Gene Spafford, 2003 (3. Auflage), O'Reilly & Asc.
- Firewalls and Internet Security, Repelling the Wily Hacker, William R. Cheswick and Steven M. Bellovin, 1994, Addison Wesley
- Hacking Exposed (4th. Ed.), McClure, Scambray and Kurtz, 2003, McGraw-Hill (www.hackingexposed.com)
- Hacking Linux Exposed, Hatch, Lee and Kurtz, 2003, McGraw-Hill (www.hackinglinuxeposed.com)
- Hack Notes Network Security Portable Reference, Horton, 2003, McGraw-Hill
- Hack Notes Linux and Unix Security Portable Reference, Dhanjani, 2003, McGraw-Hill