

Digitale Forensik

Aufklären von Systemeintrüben am Beispiel von trojanischen Pferden und Rootkits



Wilhelm Dolle
Director Information Technology
interActive Systems GmbH

Vorlesung „Sicherheit in Rechnernetzen“
Prof. Dr. Bettina Schnor, Universität Potsdam
15. Juni 2005, Potsdam

Agenda

- **Bedrohungssituation im Überblick**
- **Trojanische Pferde und Rootkits**
- **Grundregeln der computer-forensischen Ermittlung**
- **Digitale Beweissammlung**
- **Analyse mit geeigneten Werkzeugen**
- **Zusammenarbeit mit den Ermittlungsbehörden**



Bedrohungssituation

- **Steigende Anzahl von Angriffen / Sicherheitsvorfällen**
- **Verfügbarkeit von (einfachen) Angriff-Tools**
- **Unterschiedlicher Kenntnisstand der Täter**
 - ✘ Hacker / Cracker
 - ✘ Script Kiddies
 - ✘ Innen- / Aussentäter
- **Unterschiedliche Motivation der Täter**
 - ✘ Wirtschaftliche Gründe
 - ✘ „Testen“ von technischem Know-How
 - ✘ (Industrie)Spionage



Angriffsmethoden im Überblick

● Typischer Angriffsablauf

- ✘ Footprinting (Netzstruktur aufklären)
- ✘ Protokoll- und Portscans (aktiv / passiv)
- ✘ Enumeration (Betriebssystemerkennung, Bannergrabbing, Verwundbarkeitsscans)
- ✘ Ausnutzen einer Sicherheitslücke durch Anwendung eines Exploits (z.B. Buffer Overflow)
- ✘ Verstecken und Festsetzen (Rootkit installieren, Hintertüren, Sniffer, Keylogger, ...)
- ✘ Einbruchsspuren verwischen
- ✘ System missbrauchen



Verwundbarkeiten

- <http://www.sans.org/top20/>

Top Vulnerabilities to Windows Systems

- ◆ W1 Web Servers & Services
- ◆ W2 Workstation Service
- ◆ W3 Windows Remote Access Services
- ◆ W4 Microsoft SQL Server (MSSQL)
- ◆ W5 Windows Authentication
- ◆ W6 Web Browsers
- ◆ W7 File-Sharing Applications
- ◆ W8 LSAS Exposures
- ◆ W9 Mail Client
- ◆ W10 Instant Messaging

Top Vulnerabilities to UNIX Systems

- ◆ U1 BIND Domain Name System
- ◆ U2 Web Server
- ◆ U3 Authentication
- ◆ U4 Version Control Systems
- ◆ U5 Mail Transport Service
- ◆ U6 Simple Network Management Protocol (SNMP)
- ◆ U7 Open Secure Sockets Layer (SSL)
- ◆ U8 Misconfiguration of Enterprise Services NIS/NFS
- ◆ U9 Databases
- ◆ U10 Kernel



Begriffsdefinitionen

- **Viren**
- **Würmer**

- **Hoaxes**
- **Phishing (Benutzer werden z.B. über Emails auf Webseiten gelockt)**
- **Pharming (Umleiten von Anfragen via Manipulation von DNS-Servern)**

- **Trojanische Pferde (aus dem Site Security Handbook)**
 - ✘ *„Ein tojanisches Pferd kann ein Programm sein, das etwas nützliches oder auch nur etwas interessantes tut. Es tut immer etwas unerwartetes, wie beispielsweise ohne Ihr Wissen Passwörter stehlen oder Dateien kopieren.“*

- **Rootkits**
 - ✘ Übergang zu trojanischen Pferden oft fließend

Was sind Rootkits?

- **Keine Angriffswerkzeuge**
- **Oft leichte Installation**
- **Verbergen Spuren (Dateien, Prozesse, Netzverbindungen, ...) des Einbrechers**
- **Bringen trojanisierte Systemprogramme und andere Tools mit („Admin Bausatz“)**
- **Schaffen dauerhaften Zugang zum kompromittierten System (Backdoor)**
- **Enthalten zusätzlich oft Keylogger, Sniffer, Logfilecleaner**
- **Memory-Based Rootkits / Persistent Rootkits**
- **Dateibasierte/-modifizierende Rootkits (User-Mode Rootkits)**
- **Kernelbasierte/-modifizierende Rootkits (Kernel-Mode Rootkits)**



Historie von Rootkits

- Ende 80'er: manipulieren von Logdateien
- 1989: Phrack-Magazin: Umgehen von Unix-Überwachung
- 1994: erster CERT Hinweis auf eine Sammlung von Programmen
- Erste Erwähnung von „Rootkits“ in Zusammenhang mit SunOS
- 1996: erste Linux-Rootkits
- 1997: Phrack-Magazine: LKM-Rootkits vorgeschlagen (zunächst für Linux verfügbar später auch für andere Unix-Varianten)
- 1998: Non-LKM kernel patching von Silvio Cesare
- 1997: heroin
- 1999: knark / adore LKM
- 1999: Kernel Rootkits für Windows (NT-Rootkit)
- 2000: T0rnkit v8 libproc library Rootkit veröffentlicht
- 2001 KIS, SuckIT manipulieren den Kernel direkt im Hauptspeicher (Technik 1998 beschrieben)
- Ab 2002: Sniffer Backdoors in Rootkits

Dateibasierte Rootkits

- Ersetzen bzw. verändern Systembefehle und Überwachungsprogramme im Dateisystem
- Klassische Rootkits (z.B. Irk3, Irk4, Irk5, t0rnkit)
- Unterart: Library Rootkits
 - ✘ t0rnkit tauscht zum Beispiel libproc.so aus, die dafür verantwortlich ist, dass Prozessinformationen über das /proc-Dateisystem vom Kernel an anfragende Prozesse weitergegeben werden können



Dateibasierte Rootkits – t0rnkit

- **syslogd wird angehalten**
- **Hash-Wert eines Passwortes (Hintertür) in /etc/ttyhash**
- **Trojanisierter SSH-Daemons unter /usr/sbin/nscd**
- **Start als „# Name Server Cache Daemon..“ in der Datei /etc/rc.d/rc.sysinit (Defaultport 47017)**
- **Konfigurationsdateien unter /usr/info/.t0rn**
- **Austausch von: login, ls, netstat, ps, ifconfig, top, du und find (Zeitstempel & Originalgrößen werden zurückgesetzt)**
- **in.fingerd öffnet eine Shell am Port 2555**
- **/usr/src/.puta enthält verschiedene Binarys (unter anderem einen Sniffer)**
- **telnet, rsh und finger werden in /etc/inetd.conf aktiviert**
- **syslogd wird wieder gestartet**



Kernelbasierte Rootkits

- LKM oder direkte Modifikation des Kernels im Speicher
- Vorteile:
 - ✗ Privilegierter Zugriff auf das System
 - ✗ Behandlung von Netzwerkpaketen vor lokaler Firewall (stealth Backdoor)
 - ✗ Manipulation der Systemsprungtabelle oder direkt der Systemfunktionen
 - ✗ Keine Änderung von Systemprogrammen nötig



Interessante Systemfunktionen

- **open()** – lesender Zugriff Original, ausführender Zugriff trojanisierte Datei
- **getdents(), mkdir(), chdir(), rmdir()** – Verstecken von Verzeichnissen/Dateien
- **execve(), clone(), fork()** – Ausführen von Programmen mit bestimmten Eigenschaften (verstecken), und Vererbung an Kindprozesse
- **stat()** – Manipulation der Dateieigenschaften
- **ioctl()** – Device-Kontrolle, z.B. kein promisc-Bit (Sniffer) zu sehen



Rootkits die LKM benutzen

- **Rootkits die ladbare Kernelmodule benutzen benötigen:**
 - ✘ insmod, lsmod, rmmod
- **Verbreitete Exemplare:**
 - ✘ Knark (für Kernel 2.2) – speziell zur Täuschung von Programmen wie Tripwire oder md5sum
 - ✘ Adore (für Kernel 2.2 und 2.4)
 - ✘ Module: adore.o und cleaner.o
 - ✘ Komandozeilentool: ava



Rootkits direkt über den Hauptspeicher

- **Bringen eigene Modulloader mit oder benötigen keine LKM Unterstützung**
- **Greifen direkt auf den im Hauptspeicher befindlichen Kernel über `/dev/kmem` zu**
- **Technik wurde schon 1998 beschrieben**
- **Monolithischer Kernel hilft in einigen Fällen nicht mehr**



Kernel Intrusion System (KIS)

- DefCon 9 / 2001: Kernel Intrusion System (KIS) von optyx
- Bringt eigenen Modullader mit (Kernel-Memory-Patching)
- Versteckte Hintertür: lauscht erst auf einem Port nachdem ein spezielles Paket (beliebiger Port) an den Rechner geschickt wurde (Stealth-Backdoor)
- Graphischer Client und Server
- Interface für Plug-ins (leicht erweiterbar)



NT-Rootkit


- 1999 von Greg Hoggan als Proof of Concept
- Gerätetreiber `_root_.sys`
- Startdatei `deploy.exe`
- Starten: `net start _root_`
- Stoppen: `net stop _root_`

- Eigener TCP/IP-Stack
- Telnet Backdoor unter IP 10.0.0.166 (beliebiger Port)
- Verstecken von Dateien und Prozessen
- Möglichkeit zum Umleiten von Programmaufrufen

- "Heimliche Hintertüren - Rootkits aufspüren und beseitigen"; Wilhelm Dolle, Thomas Fritzing, Jürgen Schmidt; Online-Artikel zum Start der Seite "Heise Security"; Juli 2003; <http://www.heise.de/security/artikel/38057>



NT-Rootkit (Starten und Telnet-Backdoor)



```
cmd.exe
C:\WINNT>net start _root_
_root_ wurde erfolgreich gestartet.
C:\WINNT>_

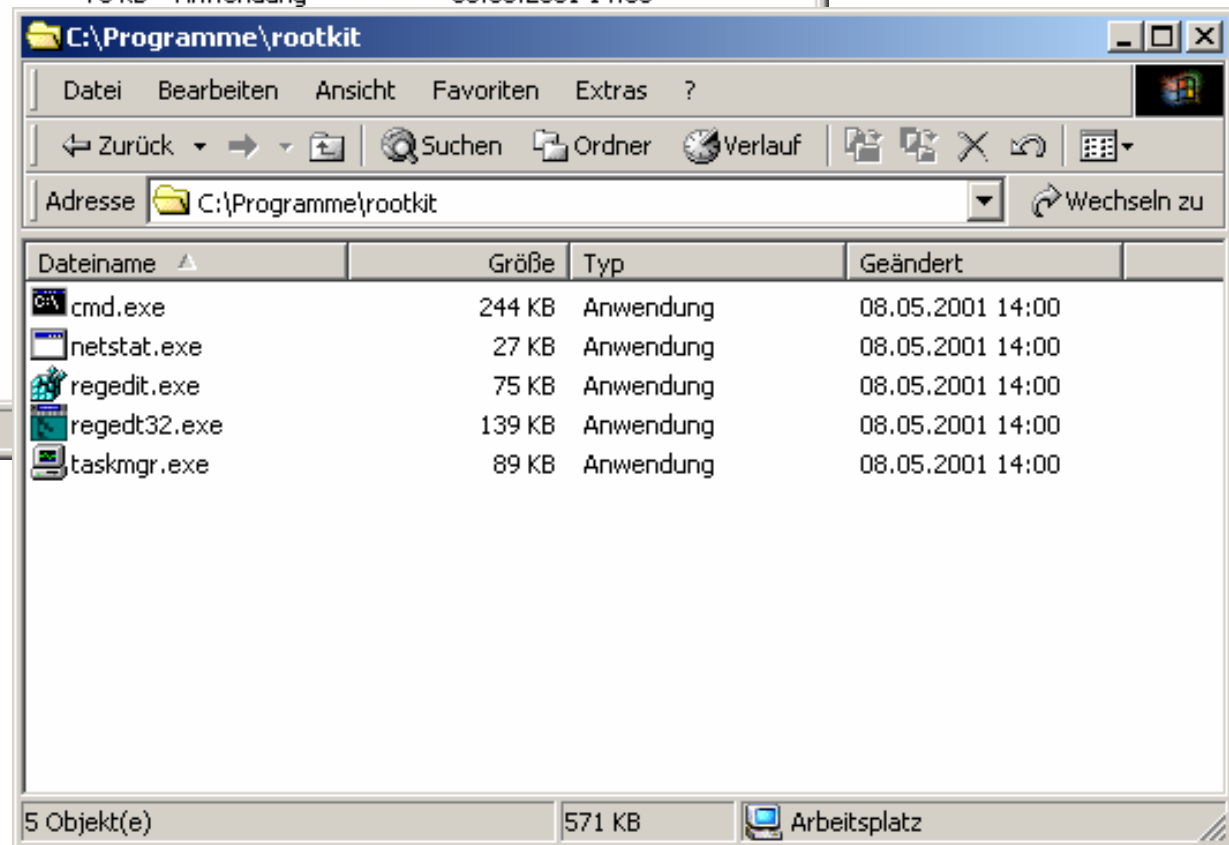
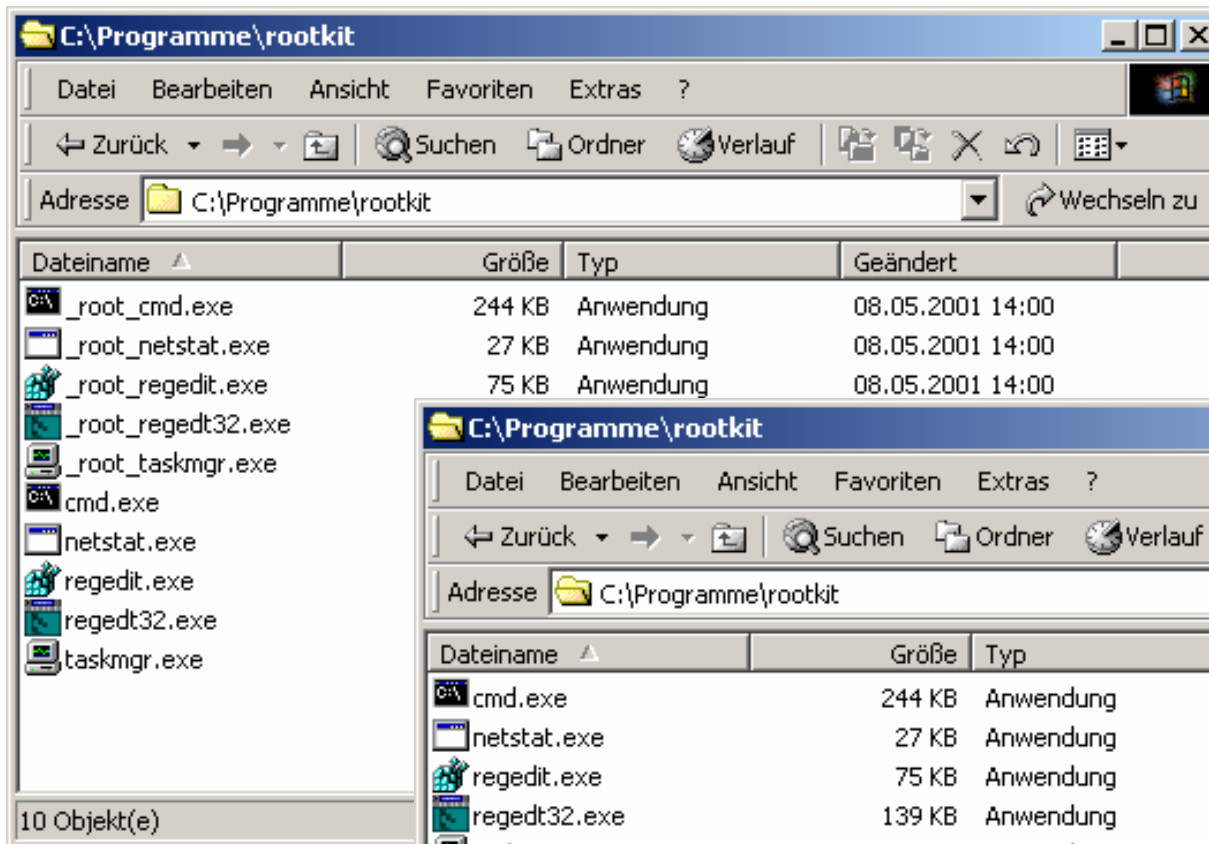
C:\WINNT\System32\cmd.exe - telnet 10.0.0.166
¿help
Win2K Rootkit by the team rootkit.com
Version 0.4 alpha
-----
command          description
ps                show proclist
help              this data
buffertest        debug output
hididir           hide prefixed file/dir
hideproc          hide prefixed processes
debugint          (BSOD)fire int3
sniffkeys         toggle keyboard sniffer
echo <string>     echo the given string

*(BSOD) means Blue Screen of Death
if a kernel debugger is not present!
*'prefixed' means the process or filename
starts with the letters '_root_'.

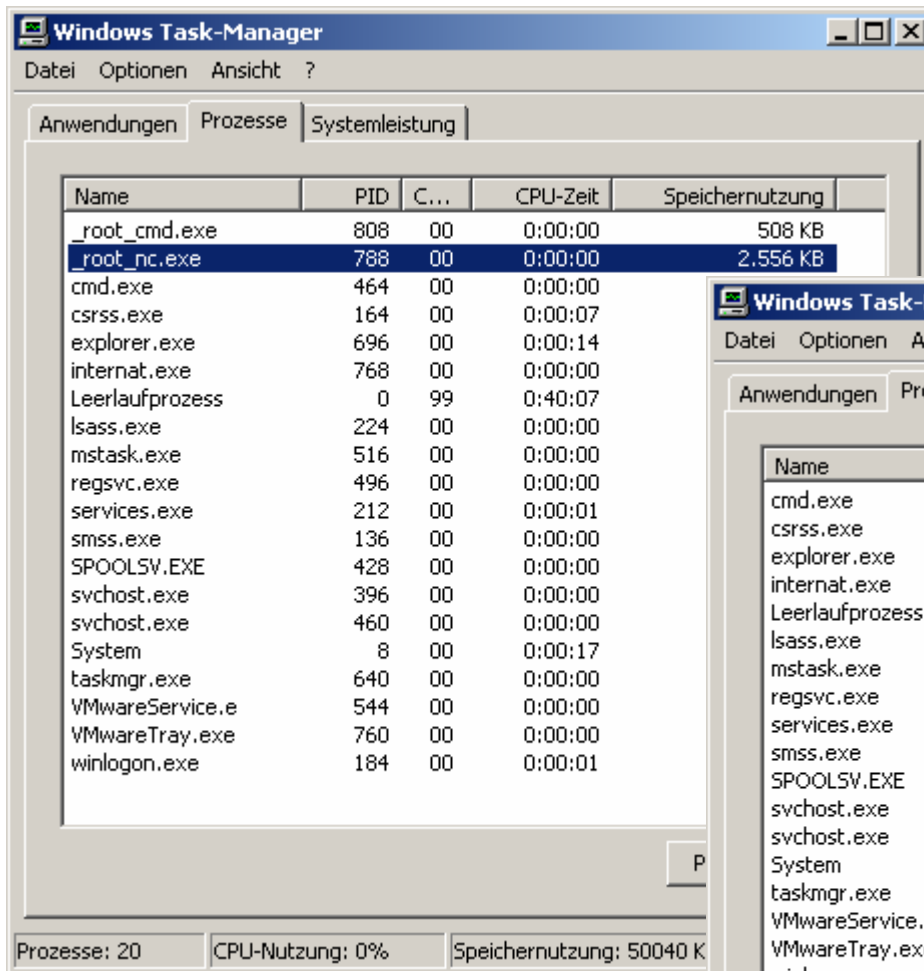
¿
```



NT-Rootkit (Dateien verstecken)



NT-Rootkit (Prozesse verstecken)

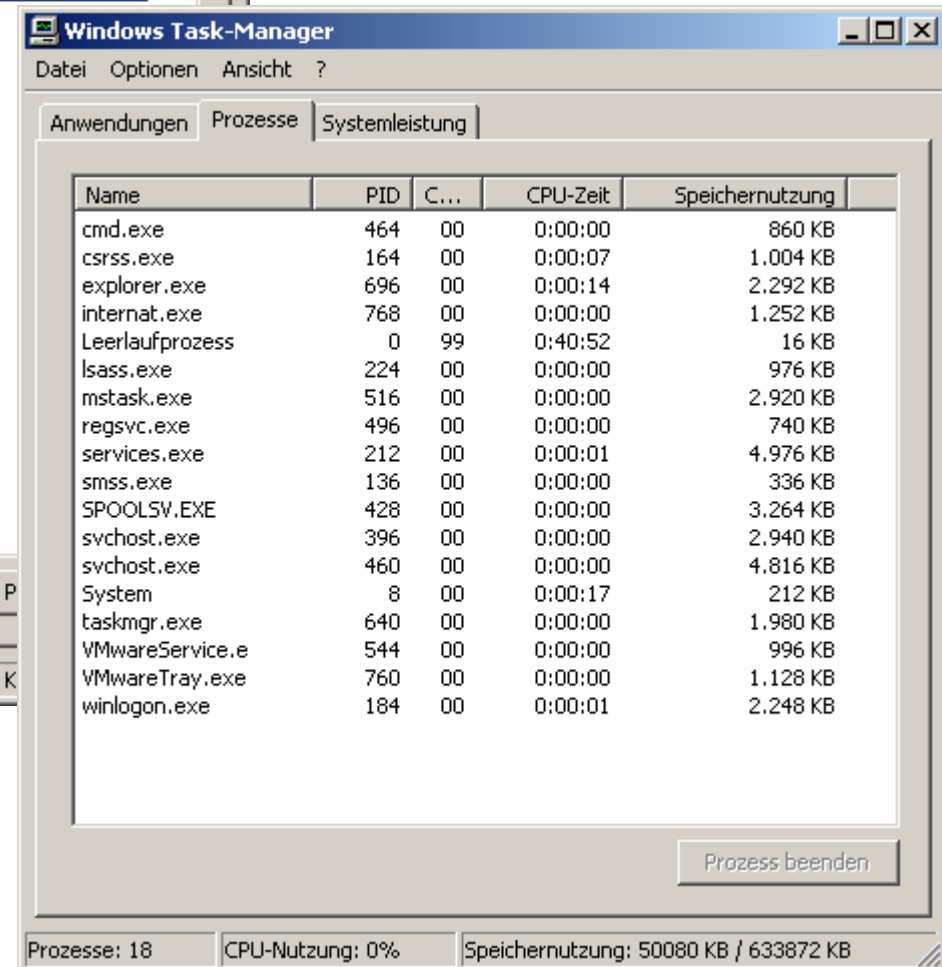


Windows Task-Manager

Prozesse

Name	PID	C...	CPU-Zeit	Speichernutzung
_root_cmd.exe	808	00	0:00:00	508 KB
_root_nc.exe	788	00	0:00:00	2.556 KB
cmd.exe	464	00	0:00:00	
csrss.exe	164	00	0:00:07	
explorer.exe	696	00	0:00:14	
internat.exe	768	00	0:00:00	
Leerlaufprozess	0	99	0:40:07	
lsass.exe	224	00	0:00:00	
mstask.exe	516	00	0:00:00	
regsvc.exe	496	00	0:00:00	
services.exe	212	00	0:00:01	
smss.exe	136	00	0:00:00	
SPOOLSV.EXE	428	00	0:00:00	
svchost.exe	396	00	0:00:00	
svchost.exe	460	00	0:00:00	
System	8	00	0:00:17	
taskmgr.exe	640	00	0:00:00	
VMwareService.e	544	00	0:00:00	
VMwareTray.exe	760	00	0:00:00	
winlogon.exe	184	00	0:00:01	

Prozesse: 20 CPU-Nutzung: 0% Speichernutzung: 50040 K



Windows Task-Manager

Prozesse

Name	PID	C...	CPU-Zeit	Speichernutzung
cmd.exe	464	00	0:00:00	860 KB
csrss.exe	164	00	0:00:07	1.004 KB
explorer.exe	696	00	0:00:14	2.292 KB
internat.exe	768	00	0:00:00	1.252 KB
Leerlaufprozess	0	99	0:40:52	16 KB
lsass.exe	224	00	0:00:00	976 KB
mstask.exe	516	00	0:00:00	2.920 KB
regsvc.exe	496	00	0:00:00	740 KB
services.exe	212	00	0:00:01	4.976 KB
smss.exe	136	00	0:00:00	336 KB
SPOOLSV.EXE	428	00	0:00:00	3.264 KB
svchost.exe	396	00	0:00:00	2.940 KB
svchost.exe	460	00	0:00:00	4.816 KB
System	8	00	0:00:17	212 KB
taskmgr.exe	640	00	0:00:00	1.980 KB
VMwareService.e	544	00	0:00:00	996 KB
VMwareTray.exe	760	00	0:00:00	1.128 KB
winlogon.exe	184	00	0:00:01	2.248 KB

Prozesse: 18 CPU-Nutzung: 0% Speichernutzung: 50080 KB / 633872 KB

Rootkits starten (Beispiele)

- **Modifikation des init-Programms**
- **Modifikation der Startscripte**
- **Modifikation von Serverprogrammen die oft beim Systemstart aufgerufen werden (sshd, httpd, ...)**



Wie fällt ein kompromittiertes System auf?

- **Intrusion Detection Systeme (host- / netzwerk-basierend)**
- **File System Integrity Checker**

- **Systemabstürze**
- **Erhöhter Netzwerkverkehr**
- **Verstoß gegen die Sicherheitsrichtlinien (nicht erlaubte Protokolle, Zugriffszeiten, ...)**
- **Dateisystem wird stärker genutzt**
- **Höhere Prozessorlast**
- **Geänderte Passwörter / neue Benutzer**



Digitale Forensik

● Forensik

- ✘ Stammt vom lat. forum = der Marktplatz, das Forum, da vormals Gerichtsverfahren, Untersuchungen, Urteilsverkündungen sowie der Strafvollzug öffentlich und meist auf dem Marktplatz durchgeführt wurden
- ✘ Das Attribut forensisch bezeichnet alles, was gerichtlichen oder kriminologischen Charakter hat, wie Rechtsmedizin, forensische Psychiatrie

● Digitale Forensik / Computer Forensik

- ✘ Allgemein: Analysieren und Aufklären von Verbrechen im Zusammenhang mit Computern



Ziele und Vorgehen bei der Ermittlung

- **Wer? Was? Wo? Wann? Womit? Wie? Warum?**
- **Ablauf in Phasen**
 - ✘ Alarmierung (z.B. durch Intrusion Detection System)
 - ✘ Vorfall verfolgen (Kosten / Nutzen abwägen – wer entscheidet?)?
 - ✘ „Tatort“ sperren
 - ✘ Beweismittel sammeln (Dokumentation!)
 - ✘ Beweismittel nachweisbar gegen (unerkannte) Modifikation schützen
 - ✘ Analyse der gesammelten Daten
 - ✘ Bewertung der Ergebnisse
 - ✘ Ergebnisse dokumentieren und präsentieren
 - ✘ Evtl. Bezeugen vor Gericht (Sachverständiger)



Grundregeln der Computer-Forensik I

- **Nichts am Tatort Verändern**
- **Nicht mit Originaldaten arbeiten**
- **Alles beweisbar machen**
 - ✘ Dokumentieren und Befolgen von „Best Practices“
 - ✘ Beweisbar machen bedeutet hier einen Richter zu überzeugen
- **Hardware-Inventur des Computers**
- **Geringe Personenzahl im Untersuchungsteam**
- **Zuerst alle Daten sammeln und erst danach mit der Analyse beginnen**



Grundregeln der Computer-Forensik II

• Wie dokumentieren?

- ✘ Schriftliche Notizen jedes(!) Schrittes auf Papier
- ✘ Nummerierte und gebundene Seiten
- ✘ Sehr wichtig ist die Zeit (reale Zeit, Betriebssystem, Hardware – Unterschiede dokumentieren)
- ✘ Zeitstempel und Unterschrift/Kürzel jedes Schrittes
- ✘ Zusätzlich: elektronische Dokumentation der auf einem Rechner ausgeführten Befehle (und deren Ergebnisse) z.B. script(1)



Digitale Beweissammlung

- **Daten in der Reihenfolge ihrer Vergänglichkeit sammeln**
 - ✘ Registerwerte, Cache-Inhalte, Hauptspeicher
 - ✘ Temporäre Dateien, Zustand des Netzwerkes, laufende Prozesse
 - ✘ Daten auf Festplatten
 - ✘ Daten auf Disketten, CD/DVD-RW, USB-Geräten, ...
 - ✘ Daten auf CD/DVD-R, Papier
- **Daten möglichst auf einen eigenen Analyserechner sichern**



Forensische Toolbox

- VOR der Untersuchung zusammengestellte bootbare CD mit Mini-Linuxsystem (bzw. dem zu untersuchenden System)
- Alle Programme statisch gelinkt
- **Typische Unix/Linux Programme:**
 - ✘ dd, cp, cat, ls, ps, lsof, strings, find, file, bash, grep, less, vi, perl, ifconfig, kill, nc/netcat, tcpdump, arp, des, df, diff, du, last, lsmdu, md5, mv, netstat, rpcinfo, showmount, top, uname, uptime, w, who, fdisk, gzip
- **Spezielle Programme zur forensischen Datensammlung und Analyse**
 - ✘ TCT, TCTUtils, The Sleuth Kit, Autopsy, cryptcat, perl
- **Spezielle (Linux-)Distributionen**

Erstkontakt

- **Keine Panik!**
- **Situationsabhängig**
 - ✘ System nicht abschalten (flüchtige Speicher, laufende Prozesse)
 - ✘ Keinen Netzwerkstecker ziehen (aktuelle Netzverbindungen)
- **Kein Backup einspielen (Analyse unmöglich, Schwachstelle nicht beseitigt, welches ist das letzte nicht kompromittierte Backup?)**



Image der Quelle

- Hash der Quelle anfertigen
- Bit-genaues 1:1 Image der Quelle erstellen
- Hash der Kopie anfertigen und mit Quelle vergleichen
- Warum 1:1 Image und keine einfache Kopie?
 - ✘ File Slack (nicht vollständig beschriebene Cluster)
 - ✘ Unallokierte Datenblöcke
 - ✘ Keine Modifikation der Zugriffszeiten (MACtimes)
 - M – mtime: Änderung am Inhalt
 - A – atime: letzter lesender Zugriff
 - C – ctime: Inode-Änderung (Rechte, Eigentümer)



Werkzeuge zum Erzeugen eines Images

- **Computer Forensics Tool Testing Program des NIST (National Institute of Standards and Technology)**
- http://www.cftt.nist.gov/disk_imaging.htm
- **Disk Image Specifications 3.1.6 (Oktober 2001)**
- **Digital Data Acquisition Tool Specification (Draft 1 von Version 4.0, Oktober 2004)**
- **Anforderungen an das Werkzeug**
 - ✘ Erstellen eines Bit-Stream-Duplikates oder eines Images einer originalen Festplatte oder einer Partition
 - ✘ Keine Änderungen an der Originalfestplatte
 - ✘ I/O-Fehler müssen mitgeloggt werden
 - ✘ Dokumentation muss vollständig und korrekt sein



Probleme beim Erzeugen eines Images

● Linux / GNU dd (2002)

- ✘ Probleme bei Quellen mit ungerader Anzahl an Sektoren (letzter Sektor wird nicht berücksichtigt)
- ✘ Bricht bei Fehlern ab, keine Fortschrittsanzeige
- ✘ Alternative: dd_rescue

● SafeBack (2003)

- ✘ Bei FAT32 werden Kontrollwerte in Sektor 1 verändert
- ✘ Bei einigen alten BIOS-Versionen werden nicht alle Zylinder einer Festplatte berücksichtigt

● EnCase (2003)

- ✘ Bei einigen alten BIOS-Versionen werden nicht alle Zylinder einer Festplatte berücksichtigt
- ✘ Probleme beim exakten Kopieren von FAT32 und NTFS



Transfer des Images auf den Analyserechner

• netcat (nc)

- ✘ Ziel: `nc -l -p 6666 >> partition1-2005-06-15.img`
- ✘ Quelle: `dd if=/dev/hda1 bs=512 | nc -w 2 [Ziel-IP] 6666`
- ✘ Kompression: `gzip -dc` bzw. `gzip -c`

• cryptcat (Verschlüsselung)

• socat

- ✘ Ziel:
`socat -u tcp4-listen:6666 create:/home/wd/partition1-2005-06-15.img`
- ✘ Quelle:
`dd if=/dev/hda1 bs=512 | socat -u - tcp4:[Ziel-IP]:6666`



Datensammlung (nicht chronologisch)

- **Hauptspeicher**

- ✘ `dd bs=1024 < /dev/kmem | netcat -w 2 [Ziel-IP] 6666`

- **Netzwerkverbindungen**

- ✘ `Netstat -an | netcat -w 2 [Ziel-IP] 6666`

- **Prozesse ohne zugehörige Binärdatei**

- ✘ Einbrecher löschen oft das Binary nach dem Ausführen

- ✘ `cat /proc/[PID]/exe > [Datei]`

- **Weitere Beispiele: last, who, w, ps, lsof, arp, ...**

- **Externe Quellen: Logserver, Firewall-, Intrusion Detection- und Router-Logfiles**



Analyse – Zu beantwortende Fragen

- Wer hat wann was auf dem System gemacht?
- Ist der Angreifer noch aktiv?
- Wie genau ist der Einbrecher ins System eingedrungen?
- Welche Spuren wurden hinterlassen?
 - ✘ Tools, Rootkits
 - ✘ Welche Programmier- / Scriptsprache?
 - ✘ Aus welchem Sprachraum kam der Angreifer?
 - ✘ Was wurde gelöscht bzw. modifiziert?
 - ✘ Versteckte / verschlüsselte Dateien / Verzeichnisse / Partitionen?



Werkzeuge

- The Coroner's Toolkit und TCT-Utills
- The Sleuth Kit
- Autopsy Forensic Browser

- EnCase



System untersuchen

- Wichtige Dateien überprüfen (inetd.conf / xinetd, services, Startdateien in rc.d, ...)
- Oft „trojanisierte“ Programme
 - ✘ ps, ls, find, ifconfig, netstat, du, df
 - ✘ sshd, httpd
 - ✘ login, passwd, inetd, tcpd
- Verdächtige MACtimes in /sbin/ oder /usr/sbin? (verdächtige Binaries mit „strings“ ansehen)
- Dateien mit Link Count kleiner 1 (ls -l +L1)
- Historys ansehen (z.B. .bash_history)
- Normale Dateien im Device File System (find /dev -type f)



System untersuchen (cont.)

- Geladene Module prüfen
- Nach Interfaces im „promiscuous mode“ suchen (Sniffer)
- Analyse der installierten Pakete mit RPM

```
# rpm --verify --all  
S.5..... /bin/lS  
S.5....T /usr/bin/named  
S.5..... /bin/netstat
```

- Logfiles auf verdächtige Einträge durchsuchen; Beispiel:
Oct 31 16:52:33 Opfer telnetd: connect from x.x.x.x
Oct 31 16:52:34 Opfer telnetd: ttloop: peer died: ...
=> Buffer-Overflow-Angriff auf telnetd



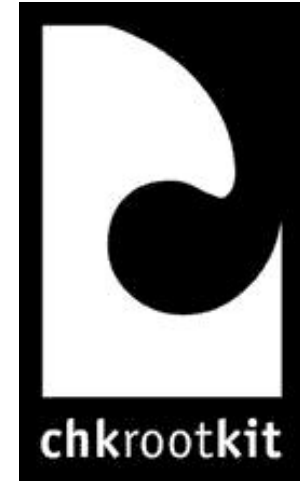
Rootkits finden (Konzepte)

- Offene (versteckte) Ports (Scan von außen mit netstat vergleichen – hidden Backdoor?)
- Versteckte Dateien
- Modulliste (wenn nicht versteckt)
- Vergleich der Systemsprungtabelle mit System.map
- Vergleich der Kernelsymboltabelle (/proc/ksyms) mit der eines „sauberen“ Kernels
- Signatur (wenn bekannt und nicht verschlüsselt) im Speicher suchen (z.B. strings /dev/kmem)
- PID-Test (versuchen alle „freien“ PIDs durch einen Testprozess zu belegen)

ChkRootKit

- www.chkrootkit.org
- Lokal ausführbares Script (getestet auf Linux, BSD, Unix)
- Erkennt nahezu alle bekannten Rootkits (Stand Februar 2005, Version 0.45, ca. 60)
- Benutzt lokal vorhandene Befehle
- Analoge Probleme wie mit ifconfig bei der Erkennung des promisc-Modus (besser ip link show)
- Evtl. Einsatz von einem schreibgeschützten Medium mit allen notwendigen (statisch gelinkten) Befehlen:

```
chkrootkit -p /mnt/cdrom/bin
```



ChkRootKit Ausgabe

```
# ./chkrootkit
```

```
...
```

```
Checking 'su' ... not infected
```

```
Checking 'ifconfig' ... INFECTED
```

```
...
```

```
Checking 'identd' ... not found
```

```
Checking 'init' ... INFECTED
```

```
...
```

```
Searching for Adore Worm... nothing found
```

```
...
```

```
Checking 'sniffer' ...
```

```
eth0 is not promisc
```



Rootkit Hunter

- www.rootkit.nl
- **Benutzt u. a. die folgenden Methoden**
 - MD5 hash Vergleich
 - Suche nach typischen Dateimanipulationen von Rootkits
 - Falsche Berechtigungen bei Binärdateien
 - Suche nach verdächtigen Zeichenketten in LKM
 - Suche nach versteckten Dateien
 - Optional scannen innerhalb von Plaintext- und Binärdateien
- **unterstützt Linux und BSD (auch andere Unixe)**
- **Erkennt nahezu alle bekannten Rootkits / trojanischen Pferde**



Rootkits finden unter Windows (2005)

● RootkitRevealer von SysInternals

- ✘ <http://www.sysinternals.com/Utilities/RootkitRevealer.html>
- ✘ Liefert Hinweise auf Anomalien im System (versteckte Dateien)
- ✘ Vergleicht High-Level Zugriffe auf Dateien und die Registry über die Windows-API mit dem Low-Level Zugriff direkt auf ein FAT- bzw. NTFS-Dateisystem
- ✘ Erkennt alle auf www.rootkit.com bekannten Rootkits (z.B. AFX, Vanquish, HackerDefender)

● Strider GhostBuster Rootkit Detection von Microsoft

- ✘ <http://research.microsoft.com/rootkit/>
- ✘ Juni 2005: noch nicht veröffentlicht
- ✘ Bietet 3 Modi: WinPE CD-Boot, Inside-the-box, User-Mode
- ✘ Findet versteckte Dateien, Prozesse und Registry-Keys



STRIDER



Entfernen von Rootkits

- Backup einspielen erst nach digitaler Forensik (wann war das letzte „saubere“ Backup?)
- Manipulierte Dateien durch Originale ersetzen
- Module entfernen
- Startscripte säubern / Startpunkte des Rootkits entfernen
- Wenn möglich: System komplett neu aufsetzen!



Proaktive Schutzmaßnahmen

- Gut ausgebildete Systembetreuer (sowie ausreichend Zeit)
- Verhindern des Ausnutzens von Buffer-Overflows und Format-String Angriffen
- Auditing von in den Kernel eingefügten Modulen (Modifikation von insmod)
- Problem: der allmächtige Administrator
 - ✘ LIDS (Linux Intrusion Detection System)
 - ✘ Security-Enhanced Linux (ursprünglich von der NSA)
 - ✘ GrSecurity
 - ✘ RSBAC (Rule Set Based Access Control)



Datenschutz

- Datenvermeidung
- Datensparsamkeit
- Anonymisierung
- Pseudonymisierung



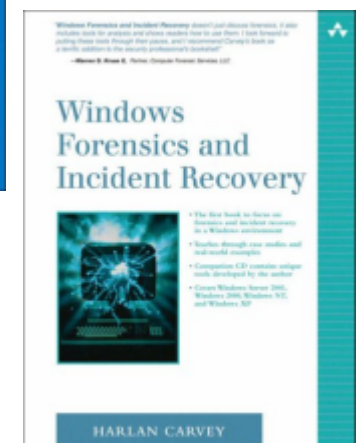
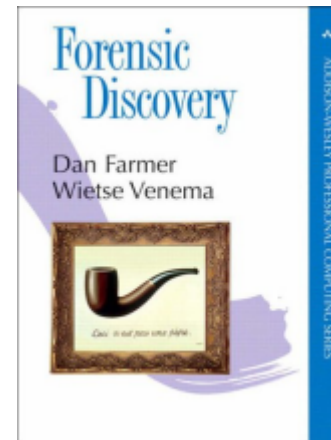
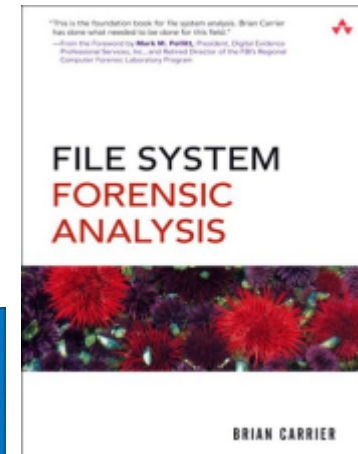
Zusammenarbeit mit Ermittlungsbehörden

- **Wer trifft die Entscheidungen?**
- **Zivilrechtliches vs. Strafrechtliches Vorgehen**
- **Strafanzeige**
 - ✘ Wird eine Strafanzeige gestellt?
 - ✘ Wer ist berechtigt eine Strafanzeige zu stellen?
 - ✘ Einfluss auf ein laufendes Strafverfahren?
- **Tatortprinzip**
 - ✘ Wo sitzt der Täter?
 - ✘ Wo entstand der Schaden (Hilfstatort)?
- **Sammlung von Beweisen (eigenes Team oder Ermittlungsbehörden)?**



Literatur

- **Computer-Forensik, Alexander Geschonneck, dpunkt, 2004**
- **File System Forensic Analysis, Brian Carrier, Addison-Wesley, 2005**
- **Forensic Discovery, Daniel Farmer, Wietse Venema, Addison Wesley, 2005**
- **Windows Forensics and Incident Recovery, Harlan Carvey, Addison-Wesley, 2004**



Fragen?

Vielen Dank für die Aufmerksamkeit

Folien TCPA und Digitale Forensik

<http://www.dolle.net>

Wilhelm Dolle, CISA, CISSP, IT-Grundschutz-Auditor
Director Information Technology

iAS interActive Systems GmbH
Dieffenbachstrasse 33c
D-10967 Berlin

phone +49-(0)30-69004-100
fax +49-(0)30-69004-101
mail wilhelm.dolle@interActive-Systems.de
web <http://www.interActive-Systems.de>

