Digitale Forensik unter Linux

Grundlagen, Datensammlung und Analyse

Wilhelm Dolle, Head of Networking Unit & IT-Security (wilhelm.dolle@interActive-Systems.de)
www.interActive-Systems.de/security

Was ist Digitale Forensik?

- Spurensicherung nach der Tat
- Analyse
  - Welche Vorgänge ...
  - ... haben wann ...
  - ... und wie zum erfolgreichen Einbruch geführt?
  - Wer?
- Dokumentation / Beweissicherung (für eventuelle juristische Verfolgung)

Nach einem erfolgreichen Angriff

- Was geschieht nach einem erfolgreichen Einbruch?
  - Überwachungsmethoden analysieren
  - Verwischen von Spuren / Entdeckung vermeiden
  - Installation eines Rootkits
  - Einrichtung von Backdoors
  - Angriff auf weitere Systeme

Wie fällt ein kompromittiertes System auf?

- Intrusion Detection Systeme (host- / netzwerkbasierend)

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

Grundsätze der digitalen Forensik

- *Secure and isolate the scene. Record the scene. Conduct a systematic search for evidence.* (aus Saferstein, *An Introduction to Forensic Science*)

- Änderungen am Originalsystem vermeiden
- Uhrzeit/Datum notieren (Realzeit zu Systemzeit)
- Änderungen dokumentieren (mit Uhrzeit)
- Hardware-Inventur des Originalsystems
- Geringe Personenzahl im Untersuchungsteam
- Zuerst Daten sammeln - dann erst analysieren
- Niemals mit Originaldaten arbeiten

Momentaufnahme des Systems

- Daten in der Reihenfolge ihrer Vergänglichkeit sichern:
  - Registerwerte, Cacheinhalte
  - Hauptspeicher
  - Aktueller Zustand des Netzwerkes
  - Laufende Prozesse
  - Daten auf Festplatten
  - Daten auf Disketten, CD-RW, Streamer, ...
  - Daten auf CD-R, Papier, ...

- **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, lsmod, md5, mv, netstat, rpcinfo, showmount, top, uname, uptime, w, who, fdisk, gzip
  - Spezielle Programme zur forensischenDatensammlung und Analyse
    - TCT, TCTUtils, Autopsy, cryptcat, perl

- Keine Panik!

- System nicht abschalten (flüchtige Speicher, laufende Prozesse, Gefahren beim Hoch- und Runterfahren – logische Bomben)
- Keinen Netzwerkstecker ziehen (aktuelle Netzverbindungen)
- Kein Backup einspielen (Analyse unmöglich, Schwachstelle nicht beseitigt, welches ist das letzte nicht kompromittierte Backup?)

Dokumentation der Datensammlung

- Jeden Schritt dokumentieren (während der Datensammlung nicht erst danach)
- Notizbuch mit nummerierten Seiten
- Uhrzeit der Ausführung

- Textein- und ausgaben mitloggen
- Evtl. graphische Ausgaben per Screenshots sichern (z.B. mit xwd)

- Datensammlung möglichst auf eigenen Analyserechner

Dokumentation mit script und netcat

- Alle Aktivitäten auf der Konsole mitloggen:

  mkfifo named_pipe

  script –f named_pipe

  cat named_pipe | nc Host Port

- netcat (Schweizer Messer des Netzwerkad-ministrators)
  - Client und Server für UDP und TCP
  - Beliebige Ports wählbar
  - Telnet-Funktionalität

Mit netcat im Netzwerk

- Über das Netzwerk per netcat
  - Ziel: nc –l –p 6666 >> Datei
    nc –l –p 6666 | gzip >> Datei
  - Quelle: [Daten] | nc –w 2 [Ziel-IP] 6666
- Bei nicht vertrauenswürdigen Netzen verschlüsseln mit des ...
  - nc –l –p 6666 | des –d –c –k [Schlüssel] >> Datei
  - [Daten] | des –e –c –k [Schlüssel] | nc –w 2 [Ziel-IP] 6666
- ... oder Benutzung von cryptcat

Datensammlung (Hauptspeicher/Prozesse)

- Hauptspeicher
  - Ziel:

    nc –l –p 6666 > mem.img
  - Quelle:

    dd if=/dev/mem | nc –w 2 [Ziel-IP] 6666

- Prozesse
  - Über ps:

    ps enf -Aelf --cols 1000
  - Über /proc:

    ls –lR /proc/[0-9]*

Datensammlung (Netzwerkzustand)

- Konfiguration der Netzwerkkarten
  - ifconfig –a
  - netstat –iea
  - ip addr show (iproute2-Utilities)
- Routingtabellen
  - route –n und route –Cn
  - netstat –rn
  - ip route show table main (iproute2-Utilities)
  - ip rule show (iproute2-Utilities)
- Paketfilterregeln?
  - iptables –L –vn --line-numbers

Datensammlung (offene Dateien)

- Offene Dateien

  lsof
- Offene Netzwerkverbindungen

  lsof -i
- Alle Sockets

  lsof –U
- Gelöschte aber noch offene Dateien

  lsof +L1

Prozesse ohne zugehörige Datei

- Einbrecher können das Binary nach dem Ausführen löschen
- Einbrecher können so zum Beispiel Sniffer-Daten verstecken

- Wiederherstellung bei installiertem proc-Filesystem
  - cat /proc/[PID]/exe > [Datei]

Datensammlung (weitere Infos)

- last (Wer war zuletzt eingeloggt?)
- who (Wer ist eingeloggt?)
- w (Wer ist eingeloggt und was macht er?)
- arp (MAC-Adressen im Cache)
- fdisk (Partitionierungsschema)
- ....

Datensammlung (Festplatteninhalt)

- Bitstream Image einer Partition

- Ziel:

  nc –l –p 6666 >> hda1.img

  nc –l –p 6666 | gzip >> hda1.img.gz

- Quelle:

  dd if=/dev/hda1 | nc –w 2 [Ziel-IP] 6666

Vorteile des Images gegenüber Backup

- File Slack (restlicher Speicherplatz bei nicht vollständig beschriebenen Clustern; kann Teile von überschriebenen Dateien enthalten)
- Unallokierte Datenblöcke (evtl. gelöschte Dateien)
- Keine Modifikation der Zugriffszeiten („MACtimes“ bei Unix Systemen)
  - M – mtime: Änderung am Inhalt
  - A – atime: letzter lesender Zugriff
  - C – ctime: Änderungen am Inode (Rechte, Eigentümer)

Externe Datenquellen

- Dedizierte Logserver
- Firewall-Logs
- Router-Logs
- Intrusion Detection Systeme
  - netzwerkbasierte (z.B. Snort)
  - hostbasierte
  - System Integrity Verifier (z.B. Tripwire)

Analyse (1)

- Wichtige Dateien überprüfen (/etc/passwd, /etc/group, inetd.conf / xinetd, services, Startdateien in rc.d, ...)
- Verdächtige MACtimes in /sbin/ oder /usr/sbin? (verdächtige Binaries mit „strings“ ansehen)
- Geladene Module prüfen
- Per ifconfig nach Interfaces im „promiscous mode“ suchen (Sniffer)
- Historys ansehen (z.B. .bash_history)

Analyse (2)

- Analyse der Dateisysteme mit RPM

  rpm --verify --all

- 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

- Oft leichte Installation
- Verbergen Spuren (Dateien, Prozesse, ...) des Einbrechers
- Bringen trojanisierte Systemprogramme und andere Tools mit („Admin Bausatz“)
- Schaffen dauerhaften Zugang zum kompromittierten System (Backdoor)

- Dateibasierte/-modifizierende Rootkits
- Kernelbasierte/-modifizierende Rootkits

- Klassische Rootkits (z.B. lrk3, lrk4, lrk5, t0rnkit)
- Ersetzen beziehungsweise verändern Systembefehle und Sicherheitsprogramme

- Weit verbreitetes Rootkit für Unix und Linux
- Ersetzt unter anderem die Programme du, find, ifconfig, ls, netstat, ps, top und login
- Über Konfigurationsdatei anpassbar (z. B. /dev/.hidden/psconf)
- login: bei bestimmten IP-Adressen root-Zugang ohne Passwort und Logging möglich
- Trojanisierte Programme alle 31.336 Bytes groß
- In Standardinstallation TCP-Port 47017 offen

- LKM oder direkte Modifikation des Kernels im Speicher

- Vorteile:
  - Privilegierter Zugriff auf das System
  - Behandlung von Netzwerkpaketen vor lokaler Firewall
  - Manipulation der Systemsprungtabelle oder direkt der Systemfunktionen
  - Keine Änderung von Systemprogrammen nötig

Interessante Systemfunktionen

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

Rootkits die LKM benutzen

- Erstes Auftauchen vor etwa 5 Jahren
- Rootkits benutzen ladbare Kernelmodule (benötigen: insmod, lsmod, rmmod)

- Verbreitete Exemplare
  - Knark (für Kernel 2.2)
  - Adore (für Kernel 2.2 und 2.4)

- 2001: KIS
- 2002: SucKIT

- Benötigen keine LKM Unterstützung
- Greifen direkt auf den im Hauptspeicher befindlichen Kernel über /dev/kmem zu

Kernel Intrusion System (KIS)

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

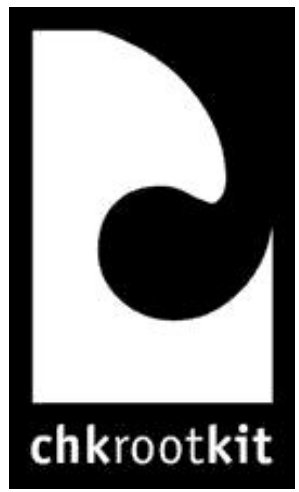
Rootkits starten

- Modifikation der Startscripte
- Austausch von Serverprogrammen (sshd, httpd, ...)

Rootkits finden (Konzepte)

- Offene Ports (Scan von außen mit netstat vergleichen – hidden Backdoor?)
- Modulliste (wenn nicht versteckt)
- Vergleich der Systemsprungtabelle mit System.map
- 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)

- Beliebte Plätze für Rootkits
  - /dev/, /usr/man, /lib/modules, /usr/lib
- Oft „trojanisierte“ Programme
  - ps, ls, find, ifconfig, netstat, du, df
  - sshd, httpd
  - login, passwd
  - inetd, tcpd



- www.chkrootkit.org (erkennt sehr viele Rootkits auf lokalem Rechner automatisch)

Hilfsmittel für die Analyse

- Kommerziell
  - z.B. EnCase
- Freeware
  - The Coroner‘s Toolkit (1999, Dan Farmer und Wietse Venema)
  - TCT-Utils (2001, Brian Carrier)
  - Autopsy (2001, Brian Carrier)
  - TASK (2002, Brian Carrier)

Ein wenig Literatur

- TCP/IP Illustrated Volume 1; W. Richard Stevens; Addison Wesley; 1994; ISBN 0-201-63346-9
- Linux System Security; Scott Mann & Ellen L. Mitchell; Prentice Hall; 2002; ISBN 0-13-015807-0
- Linux Firewalls; 2. Auflage, Robert L. Ziegler; Markt & Technik; 2002; ISBN 3-827-26257-7
- Network Intrusion Detection. An Analyst's Handbook; Second Edition; Stephen Northcutt & Judy Novak; New Riders; 2001; ISBN 0-7357-1008-2
- Intrusion Detection für Linux-Server; Ralf Spenneberg; Markt & Technik, 2002