Unix/Linux Security Response Cookbook

Don Murdoch, CISSP
GCWN, GCUX, GCIH, GCIA
MCSE & D
Information Systems Security Officer
Old Dominion University

agenda

- Set the stage for Incident Response
- Gather the necessary tools
- Discuss the phases of IR
- Provide a “minimum set” cookbook of response tools and techniques
  - Internal (on system)
  - External (off system)

resources and references

- Source material for this presentation include:
  - SANS GCUX and GCIH courses (highly recommended)
  - RFC 2350 – Site Security Handbook
  - www.giac.org practicals – a great source of real world information
  - “Incident Response and Computer Forensics”, Second Edition by Chris Prosise
  - “Guide to Computer Forensics and Investigations” Phillips, Nelson, Enfinger, Steuart

set the stage

- What do they do on CSI:XYZ?
  - Prepare
  - Collect and handle
  - Inspect and analyze Evidence
  - Reconstruct the event
  - Record, report and testify
- What do we do as computer security incident handlers?
  - Prepare
  - Identify
  - Contain
  - Eradicate
  - Recover
  - Conduct a Lessons Learned meeting

preparation phase

getting ready to handle an incident

preparation: tools – what do I need in the jump kit? (1)

- Hardware
  - dual boot laptop
  - sanitized disk(s)
  - tape backup
  - CD-R (not RW)
- Software
  - legal operating systems and tools!
  - system binaries and libraries
  - analysis tools
  - Knoppix and FIRE
  - binaries appropriate to your O.S that are statically linked if at all possible

preparation: tools – what do I need in the jump kit? (2)

- Otherware
  - Sealable bags
  - Indelible ink markers
  - Log book for incident details
  - camera
  - plan/procedures how to communicate with your staff on the issues
  - user education about "stuff you just don't do" like asking for passwords in an email or sending updates / patches in an email
- Wetware
  - understanding of your environment
  - o.s. admin skills
  - calm and restraint
    - you can't beat the attacker – be calm, cool, and collected as you respond

preparation: know your limitations

- it is easy to damage evidence
- it is even easier to misinterpret data
- if automation exists, data collection is possible but not assured – practice makes perfect
- even simple analysis can be dangerous
- ask for help

preparation: response CD

- binaries
  - arp, awk, cat, chgrp, chmod
  - chown, compress, cp, csh, cut, date, dd
  - df, diff, dig, du, echo, egrep, fdisk
  - find, finger, gzip ,head ,id, ifconfig ,ksh
  - last, lastb, ls, lsof ,ltrace, md5sum, mv
  - nc, netstat, perl ,ps, rm, route, rpm
  - script, sed, sh, strace, strings, su, tar
  - tcpdump, top, uname, vim, w, who

preparation: response CD

- libraries
  - ld-linux.so.2, libacl.so.1, libattr.so.1, libbfd-2.13.90.0.2.so
  - libc.so.6, libcrypt.so.1, libcrypto.so.2
  - libdl.so.2, libdns.so.5, libgpm.so.1
  - libisc.so.4, libm.so.6, libncurses.so.5
  - libnsl.so.1, libpam.so.0, libpcap.so.0
  - libperl.so, libproc.so.2.0.7, libpthread.so.0
  - librt.so.1, libtermcap.so.2, libutil.so.1

6/15/2004 (c) Don Murdoch; All Rights Reserved. 10

preparation: response CD

- a properly complied and recent copy of "chkrootkit"
  - from http://www.chkrootkit.org/
    - Alternatives include:
      - chkrootkit, Rootcheck:Rootkit Hunter
- required statically linked binaries
  - awk, cut, echo, egrep, find, head, id, ls, netstat, ps, strings, sed, uname

6/15/2004 (c) Don Murdoch; All Rights Reserved. 11

identification phase

identifying an incident
and
places to look for clues
and trace evidence

6/15/2004 (c) Don Murdoch; All Rights Reserved. 12

identification: what?

- slowing traffic …. or performance
  - often reported by your best sensor network – the end user
- unexplained “stuff”
  - accounts, directories, web pages, file system changes, information leakage, DoS, crashes, unusual system usage patterns
- somewhat explained “stuff”
  - IDS alarms, swatch alerts
- what time is it … ?????

identification: where?

- determine as much as you can about the local network environment
- examples include:
- perimeter
- hosts
- internal network addressing and configuration
- operating systems and installed applications

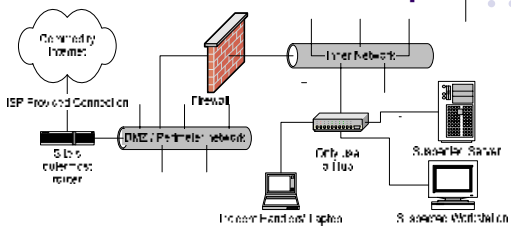
identification: perimeter

- router logs (you are logging, right?)
- firewall logs (you are logging, right?)
- i.d.s. logs (do you have an i.d.s.? snort is free after all …)
  - grep “VICTIM_IP” alert.ids
- connectivity
  - is your network connection “slow”?
  - can you see sites you normally see?
  - what is the current response time from here to there?

identification: watch and learn

- many incident handlers like to watch and learn what the attacker does *for a short period of time*
  - what are they doing
  - where are they going
  - develop an attack signature
- allowing the attacker to stay on has some potential to allow / condone the activity
- disconnecting imediately prevents any learning although it contains the incident

identification: network dumps



- monitoring with windump or tcpdump needs to be done as close to the victim as possible
- always use a hub – a switch that does not mirror traffic between the victim and the network defeats the purpose

identification: network dumps

- sudo tcpdump –s 1514 –i eth1 –w 0604_1224 –n "host 192.168.72.142"
  - -s 1514 – capture the entire Ethernet frame
  - -I eth1 – capture on your second listening interface (the one w/o an IP address!)
  - -w 06… - write a file dated with the start month, day, hour, minute
  - -n – no name resolution (faster, doesn't make outsider aware of capture)
  - "host X" – limit your scope to the victim in question; as an example 192.168.72.142

identification: syslog

- what does your central syslog system say about the victim?
  - cd /var/log/messages
  - grep VICTIM_IP messages* | grep "Nov 28“
    - where VICTIM_IP is the system in question
    - where “Nov 28” is the date in question

containment phase general

what do we generally do

containment: general

- block the attacker(s) IP and/or network
- change passwords of potentially compromised accounts / users
- determine how far reaching the attack is
- determine how far you want or need to take the case

containment phase live response

we are finished looking
we start working with the host
tread lightly…

containment: mount floppy

- record each and every command executed on the system
- mount command recording floppy (if you can)
  - # mount -n -t msdos /dev/fd0 /mnt/floppy
  - # script /mnt/floppy/basecmds.txt
  - # date
  - # history

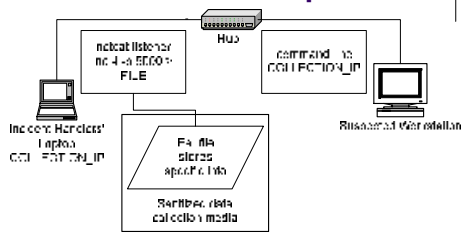
containment: mount CD for data collection

- CD
  - # mount -n /mnt/cdrom
  - # /mnt/cdrom/bin/ksh
  - # cd /mnt/cdrom/bin
  - # PATH="/mnt/cdrom/bin;"
  - # LDLIBRARYPATH="/mnt/cdrom/lib"
  - # export PATH
  - # export LDLIBRARYPATH (or LD_LIBRARY_PATH)
  - # echo $PATH
  - # echo $LDLIBRARYPATH

containment: show what you are using (may need it later)

- these commands show what you have on the floppy
- # ls -al /mnt/floppy
- # ls -al /mnt/cdrom/bin
- # ls -al /mnt/cdrom/lib

containment: data capture



- use netcat from the compromised system to the collection system
- use ONE file per “data” that is collected

containment: order of volatility

- registers, peripheral memory, caches, etc.
- memory
- network state
- running processes
- file systems
- disks
- “removable” media such as tape, CD-ROMs, DVDs, printed media, etc.
- Recently defined in RFC 3227

containment: registers, memory

- it is almost impossible to collect information on CPU registers
- it is also almost impossible to collect the contents of system memory
  - you can collect lots of state info, but memory is *always* changing as the system runs
- maybe …
  - a hibernate file can be analyzed?
- the point is to minimally impact the system
- **# ./umame -a | ./nc 192.168.16.40 5555**

6/15/2004 (c) Don Murdoch; All Rights Reserved. 28

containment: network state

- ./ifconfig | ./nc 192.168.16.40 5549
- ./netstat -a | ./nc 192.168.16.40 5550
- ./netstat -arp | ./nc 192.168.16.40 5551
- ./netstat -ap --inet | ./nc 192.168.16.40 5552
- ./route -v -n -ee | ./nc 192.168.16.40 5553
- ./arp -v -n | ./nc 192.168.16.40 5554

6/15/2004 (c) Don Murdoch; All Rights Reserved. 29

containment: logon history

- ./w | ./nc 192.168.16.40 5555
- ./last | ./nc 192.168.16.40 5556
- ./who -Hi | ./nc 192.168.16.40 5557
- ./finger -ls | ./nc 192.168.16.40 5558
- ./last -aidx | ./nc 192.168.16.40 5559
- ./lastb -aidx | ./nc 192.168.16.40 5560

6/15/2004 (c) Don Murdoch; All Rights Reserved. 30

containment: processes

- processes:
  - ./ps -auxeww | nc 192.168.16.40 5561
  - ./ps -aux | nc 192.168.16.40 5562
  - ./top -b -n1 | nc 192.168.16.40 5563
- open files
  - ./lsof -i | nc 192.168.16.40 5564
  - ./lsof -d rtd | nc 192.168.16.40 5565
  - ./lsof +M -i | nc 192.168.16.40 5566
- if you have a suspect … (example of 1236)
  - ./ls -la /proc/1236 > 192.168.16.40 5567
  - ./lsof -p 1236 | nc 192.168.16.40 5568

containment: collect log files

- ./nc 192.168.16.40 5569 </var/run/utmp
- ./nc 192.168.16.40 5570 < /var/log/wtmp
- ./nc 192.168.16.40 5571 < /var/log/messages
  - grab other syslog files ….
- ./nc 192.168.16.40 5572 < APP_SPECIFIC_LOG_FILE_HERE

containment: system files

- capture a variety of system files if you think you need to
  - ./nc 192.168.16.40 5576 < /etc/passwd
  - ./nc 192.168.16.40 5577 < /etc/shadow
  - ./nc 192.168.16.40 5578 < /etc/inittab

containment: other

- what is the state of the rpm database?
  - ./rpm -Va | nc 192.168.16.40 5579

containment: filesystem decision point

- how will you collect file access times?
- MAC times - they are ephemeral
  - m: last time modified
  - a: last time accessed
  - c: last time attributes changed (owner, permis)
- method one
  - mac-daddy or mac-robber
  - ./grave-robber -m /directory-tree
  - ./mactime 4/5/2000

containment: filesystem method two

- the file system is more volatile that the disk because one can delete files that remain on the disk
- order is critical
  - cd / (this command nees to be executed from the root)
  - /mnt/cdrom/bin/ls -laRu | nc 192.168.16.40 5573
  - /mnt/cdrom/bin/ls -alRc | nc 192.168.16.40 5574
  - /mnt/cdrom/bin/ls -alR | nc 192.168.16.40 5575
  - cd /mnt/cdrom/bin (change back to collection directory)
- on a typical system, this will generate 8 – 10 MB of data per command

containment: disconnect

- once you have collected volatile data…
  - unplug
  - power down
  - put in clean disks
  - make a set of image copies
    - original - preserve
    - one – analysis
    - two – verify the analysis
    - three – return to service
    - four – your spare so you don't have to image again

containment: disks (1)

- prepare to make a forensically sound duplicate w/ FIRE or Knoppix
- you may need to load a specialized SCSI driver
  - insmod /mnt/fire/lib/modules/2.4.20-Fire/kernel/drivers/scsi/BusLogic.o
- ideally, you would have a "write blocker"
- DD off each filesystem
  - [root@FIRE] /dev> dd if=/dev/sda of=/dev/sdb
  - 12578894+0 records in
  - 12578894+0 records out

containment: disks (2)

- verify each partition (this is an example)
- [root@FIRE] /dev>
  - for prt in '/dev/sda1' '/dev/sdb1' '/dev/sda2' '/dev/sdb2' '/dev/sda3' '/dev/sdb3'; do md5sum $prt; done
- Output
  - a5deb0419115fc58b652d442058160ba /dev/sda1
  - a5deb0419115fc58b652d442058160ba /dev/sdb1
  - b17c8b88c740631bfa7a3fa47000c6fc /dev/sda2
  - b17c8b88c740631bfa7a3fa47000c6fc /dev/sdb2
  - 3b9ab2a9215492e90ac554b9d50c464f /dev/sda3
  - 3b9ab2a9215492e90ac554b9d50c464f /dev/sdb3

containment: disk analysis

- mount and examine the COPY
  - mkdir /mnt/sdb1
  - mount -n -o noatine,nosuid,nodev,noexec,ro /dev/sdb1 /mnt/sdb1
  - mkdir /mnt/sdb2
  - mount -n -o noatine,nosuid,nodev,noexec,ro /dev/sdb2 /mnt/sdb2
- check for a common rootkit
  - # ./chkrootkit -r /mnt/sdb2

containment: analysis

- check for setuid files
  - find /mnt/sdb2/* \(-perm +004000 \) -type > /mnt/floppy/setuidfl
- check for setgid files
  - find /mnt/sdb2/* \(-perm +002000 \) -type > /mnt/floppy/setgidfl
- Search for files that *have changed since the time you suspect the incident happened*
  - touch -m 11280000 /tmp/tstmp
  - find /mnt/sdb2/* -newer /tmp/tstmp -type f -printf "%Ar %Tc %p\n" > /mnt/floppy/newfiles

eradication phase

getting the interloper off of your system

eradication: decisions

- can you remove / repair the damage?
- can you backup critical / important data?
- what was the root cause?
- how did they get in and get around?
- how far did they go?
- how far back can or should we restore data?
- rebuild or repair?

eradication: defend the castle

- patch / update other systems
- assess environment (nmap, nessus, survey) and update potentially affected software
- change network configuration to better defend the network
- review firewall and IDS rules

recovery phase

recovery

- this section is about being sure that you can return to a valid state of operation
- decisions, decisions
- monitor for the attacker to return
- monitor the system
- look for other attacks
- don't give up as criminals often return to the scene of the crime
- decrease your threat plane

lessons learned phase

gosh … glad that's over .. what do we do next ???

lessons learned

- basically – perform a post mortem analysis of the overall incident and improve operations
- avoid fingerprinting and blaming people – that is usually not constructive

for more information

- SANS – great security training
- GIAC – great practical assignments by people pursuing “hard skills” certification
- cve.mitre.org – canonical list and dictionary of security issues
- reputable security sites
  - www.securityfocus.com
  - www.cert.org
  - www.linuxsecurity.com
  - http://www.insecure.org/tools.html - top 75 security tools

for more information

- http://www.opensourceforensics.org/tools/unix.html