Host Security Exercise

Security Topics

Contents

1 Introduction

These exercices demonstrate some of the tools used for tasks that every system administrator should perform when installing or hardening a system.

2 Goals

- Learn to figure out which services are running
- Disable unnecessary services
- Scan ports to see how the machine is seen by others
- Configure automatic updates
- Use file integrity and rootkit checking tools to detect possible compromises
- Install a tool to keep a log of executed commands

3 Notes

- Commands preceded with "$" imply that you should execute the command as a general user - not as root.
- Commands preceded with "#" imply that you should be working as root.
- Commands with more specific command lines (e.g. "RTR-GW>" or "mysql>") imply that you are executing commands on remote equipment, or within another program.

4 Let's install a few tools first

```
# apt-get install lsof
```

5 What's running?

First you can see what is running on your machine by typing something like:

```
$ ps auxwww
```

You will see lots and lots of stuff go by. So, let's look at this a bit more closely:

```
$ ps auxwww | less
```

(press [spacebar] to go one page down, and [b] to go one page up)

Now, browsing through all this we can see there are a bunch of initial system processes that start to support our hardware (items in "[ ]") as well as lots of processes associated with the Gnome Display Manager (gdm and gnome). Let's filter all of this out and see what we are left with:

```
$ ps auxwww | grep -v "\[" | grep -v gdm | grep -v gnome
```

(Hint: You might want to copy and paste this in to a command window)

What's left?

Have a look and see if you can identify everything in the remaining list. Your list of processes should look something like:

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   2644  1552 ?        Ss   Jun29   0:01 /sbin/init
root       429  0.0  0.0   2152   756 ?        S    Jun29   0:00 upstart-udev-bridge --daemo
root       431  0.0  0.0   2624  1020 ?        S<s  Jun29   0:00 udevd --daemon
root       613  0.0  0.0   2620   984 ?        S<   Jun29   0:00 udevd --daemon
root       614  0.0  0.0   2620   984 ?        S<   Jun29   0:00 udevd --daemon
root       780  0.0  0.0   1852   548 ?        Ss   Jun29   0:00 dd bs=1 if=/proc/kmsg of=/v
syslog     782  0.0  0.0  33832  1748 ?        Sl   Jun29   0:00 rsyslogd -c4
104        803  0.0  0.0   2860  1104 ?        Ss   Jun29   0:00 dbus-daemon --system --fork
105        806  0.0  0.1   5352  3280 ?        Ss   Jun29   0:00 hald --daemon=yes
root       824  0.0  0.1  19412  2932 ?        Ssl  Jun29   0:00 /usr/sbin/console-kit-daemo
root       887  0.0  0.0   3344  1196 ?        S    Jun29   0:00 hald-runner
root       975  0.0  0.0   1704   552 tty4     Ss+  Jun29   0:00 /sbin/getty -8 38400 tty4
root       991  0.0  0.0   1704   548 tty5     Ss+  Jun29   0:00 /sbin/getty -8 38400 tty5
root       995  0.0  0.0   3420  1140 ?        S    Jun29   0:00 hald-addon-input: Listening
root       996  0.0  0.0   3420  1152 ?        S    Jun29   0:00 hald-addon-storage: polling
root      1001  0.0  0.0   1704   552 tty2     Ss+  Jun29   0:00 /sbin/getty -8 38400 tty2
root      1002  0.0  0.0   1704   556 tty3     Ss+  Jun29   0:00 /sbin/getty -8 38400 tty3
root      1005  0.0  0.0   5364  1100 ?        Ss   Jun29   0:00 /usr/sbin/sshd
root      1006  0.0  0.0   1704   548 tty6     Ss+  Jun29   0:00 /sbin/getty -8 38400 tty6
105       1017  0.0  0.0   3264  1120 ?        S    Jun29   0:00 hald-addon-acpi: listening
root      1036  0.0  0.0   2092   880 ?        Ss   Jun29   0:00 cron
daemon    1037  0.0  0.0   1964   416 ?        Ss   Jun29   0:00 atd
root      1063  0.0  0.1   6692  2332 ?        Ss   Jun29   0:00 /usr/sbin/cupsd -C /etc/cup
root      1170  0.0  0.2   6704  4816 ?        Ss   Jun29   0:00 /usr/sbin/munin-node
root      1245  0.0  0.0   1704   552 tty1     Ss+  Jun29   0:00 /sbin/getty -8 38400 tty1
root      1278  0.0  0.1   5168  2580 ?        S    Jun29   0:00 /usr/lib/devicekit-power/de
root     10340  0.0  0.1   8588  2972 ?        Ss   00:07   0:00 sshd: root@pts/0
root     10400  0.0  0.0   4352  1872 pts/0    Ss   00:07   0:00 -bash
root     10556  0.0  0.0   2644  1024 pts/0    R+   00:13   0:00 ps auxwww
```

You can type "man" or search in Google to figure out what all this is. For instance:

```
$ man udevd
$ man hald
$ man getty
$ man cupsd
$ man atd
```

```
$ man cron
$ man sshd
```

Etc, etc.

Once you feel pretty comfortable with what's running on your system you might consider if you need each item. If there is something running that is unnecessary, then you should consider uninstalling the software:

```
# apt-get remove <pkg_name>
```

or stopping the associated service:

```
# update-rc.d <pkg_service> remove
```

The next step is to see if any of these services are listening to the network for connections:

```
# lsof -i
```

You'll see something like:

```
COMMAND     PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd       1005 root    3u  IPv4   5150      0t0  TCP *:ssh (LISTEN)
sshd       1005 root    4u  IPv6   5152      0t0  TCP *:ssh (LISTEN)
cupsd      1063 root    5u  IPv6   5318      0t0  TCP localhost:ipp (LISTEN)
cupsd      1063 root    6u  IPv4   5319      0t0  TCP localhost:ipp (LISTEN)
sshd      10340 root    3r  IPv4  18747      0t0  TCP pc4.pacnog.bluesky.as:\
ssh->noc.pacnog.bluesky.as:34634 (ESTABLISHED)
```

Again, Google and man to figure out what is going on:

```
$ man sshd
$ man cupsd
```

What's cupsd? is this necessary on every server?

Notice that sshd is listening to all incoming connection requests (the "*"). This is a typical, potential security hole.

In our case, we will leave ssh up, but we are aware they are running and need to be patched for security updates as they come out.

For example, it is a good idea to lock down sshd a bit by not allowing the root user to log in with a passwords.

As you are not printing, let's turn off the cups printing service. Do you remember how to do this?

```
# ls /etc/init.d       <-- to find the service script name
# service cups stop
# lsof -i
```

Now we only see:

```
COMMAND     PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd       1005 root    3u  IPv4   5150      0t0  TCP *:ssh (LISTEN)
sshd       1005 root    4u  IPv6   5152      0t0  TCP *:ssh (LISTEN)
sshd      10340 root    3r  IPv4  18747      0t0  TCP pc4.pacnog.bluesky.as:\
ssh->noc.pacnog.bluesky.as:34634 (ESTABLISHED)
```

To prevent this service to start when the machine is rebooted, type:

```
# update-rc.d cups remove
```

6 Scan your machine remotely using nmap

It's usually a good idea to see how your machine looks to other users.

Log in to a PC different than yours. For example:

```
$ ssh sysadm@extX
```

Make sure that nmap is installed by doing:

```
# apt-get install -y nmap
```

Now let's scan your machine using the nmap command:

```
# nmap -sV extX     [Where "extX" is _your_ VM]
```

You should see something like:

```
Starting Nmap 5.00 ( http://nmap.org ) at 2010-06-30 09:59 SST
Interesting ports on ext2.ws.nsrc.org (67.218.55.102):
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 5.1p1 Debian 6ubuntu2 (protocol 2.0)
25/tcp open  smtp    Postfix smtpd
MAC Address: 00:0F:1F:E6:62:94 (WW Pcba Test)
Service Info: Host:  pc2.pacnog.bluesky.as; OS: Linux
```

```
Service detection performed. Please report any incorrect results at \
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.76 seconds
```

This looks reasonable. The machine is exposing smtp and ssh to the world as well as the type of OS that it is running.

Now let's scan a bit more aggressively:

```
# nmap -A -T4 extX
```

Take a look at the information presented. This will take some time, but it will contain more detail.

Now, remember to log out of your classmate's PC!

```
$ exit
```

You can use nmap to scan entire networks and find all the machines and services that are running. This is what network attack scripts do - but, they usually scan for a specific port and service, then they launch an attack when they find a service that they think they can break.

Be careful with nmap! If you scan aggressively or against an entire network you will likely set off detection alarms and you could get in trouble. Let people know before you scan if you are not in charge of the remote machines.

Now read about nmap to understand what -sV, -A, -T4 and -F are doing:

```
$ man nmap
```

7 Install a filesystem integrity checker

7.1 fcheck

Let's install fcheck. This is an intrusion detection tool that is very simple to set up and is preconfigured to do most of what you want:

```
# apt-get install fcheck
```

Once this is done you can look around to see how fcheck is configured. By default Ubuntu installs and configures fcheck in a reasonable manner and you probably don't need to do anything else.

```
$ man fcheck
```

Configuration of check is in /etc/fcheck/fcheck.cfg. Let's have a look:

```
# EDITOR /etc/fcheck/fcheck.cfg
```

Read through the file to see what directories fcheck is checking, which directories are excluded, etc. The check process is run once every two hours on the 1/2 hour. You can view this by looking at:

```
$ less /etc/cron.d/fcheck
```

The text that reads:

```
30 */2 * * *
```

is telling our system cron process to run the long check command listed in the file once every 2 hours on the 1/2 hour.

Now force fcheck to run for the first time and create a database:

```
# fcheck -ac
```

Look at the baseline file that fcheck has created:

```
# less /var/lib/fcheck/fcheck.dbf
```

Now let's make a change to a file in one of the directories that fcheck is checking.

```
# editor /etc/hosts
```

Add a blank line at the end of the file. Save the file.

Now do another forced run of fcheck:

```
# fcheck -a
```

You'll see lots of stuff go by on the screen.

you see something like:

```
PROGRESS: validating integrity of /etc/
STATUS:
    WARNING: [cv-macbook] /etc/hosts
    [Sizes: 257 - 258, Times: Jul 22 21:36 2010 - Mar 14 16:10 2012]
```

This tells you that the file /etc/hosts has changed. The cron job installed by Ubuntu will e-mail these kinds of reports to you.

7.2 incrond

Inotify in the kernel can provide real-time notification of filesystem changes. Install the incron package and configure incrond to monitor important filesystems.

```
# apt-get install incron
# tail /var/log/syslog
# cd /etc/incron.d
# EDITOR globals

add the following line (one line) to the globals file:

/etc IN_MODIFY,IN_CLOSE_WRITE,IN_CREATE,IN_DELETE /usr/bin/logger -p
news.warn "$% $@/$#"
```

For a description of the syntax of incron table files, see:

```
$ man 5 incrontab
```

That's it. The changes you make to incron are updated automatically. Because incron can recognize changes, it even recognizes when you change the configuration for incron, and it updates.

Now add a file to the /etc directory:

```
# touch /etc/dog
```

Take a look at /var/log/syslog. What does it say???

```
# tail /var/log/syslog
```

From now on, any changes you make in the /etc directory will generate syslog messages.

8 Turn on automatic installation of security updates

There is a meta package called unattended-upgrades to do this. To install:

```
# apt-get install unattended-upgrades
```

That's it. Any time a security update is placed in the Ubuntu repositories it will be automatically installed on your system. You will probably want to look at how unattended-upgrades is configured.

```
# cd /etc/apt/apt.conf.d
```

This package is configured in the file 50unattended-upgrades. Let's have a look and we will make a change to the configuration:

```
# vi 50unattended-upgrades
```

Note at the very top of the file. If you were to change this:

```
// Automatically upgrade packages from these (origin, archive) pairs
Unattended-Upgrade::Allowed-Origins {
        "Ubuntu lucid-security";
//      "Ubuntu lucid-updates";
};
```

To look like:

```
// Automatically upgrade packages from these (origin, archive) pairs
Unattended-Upgrade::Allowed-Origins {
        "Ubuntu lucid-security";
    "Ubuntu lucid-updates";
};
```

then all software package updates would be installed as well. You may, or may not, want to do this. This is generally safer for user desktops than for servers.

Let's change this line:

```
//Unattended-Upgrade::Mail "root@localhost";
```

To be:

```
Unattended-Upgrade::Mail "root@localhost";
```

That way your root account will get an email when an update is installed.

Note that you can even have your machine automatically reboot if required after an update.

Save the file and exit.

That's it. If a security update is applied you will be notified.

9 Run a rootkit checker

There is a nice tool called "chkrootkit" - This is used to see if a machine has been compromised with known software kits that install once security has been breached. You can read about this software here: http://www.chkrootkit.org/

To install, do this:

```
# apt-get install chkrootkit
```

To use it, do:

```
# chkrootkit
```

You should not see anything found or infected (hopefully!). However, it's possible for the tool to give you some false positives. You can go back to the http://www.chkrootkit.org/ web site for more information in the README and FAQ pages and you should use Google. If you don't see other people reporting false positivies like yours, then you probably need to format your hard drive, reinstall and restore data from backups.

Let's do something to make chkrootkit give you a warning:

Place your ethernet interfaces in to promiscuous mode (i.e. it listens for *all* packets on the network, not just packets coming to your machine).

```
# ifconfig lo promisc
```

Now let's re-run chkrootkit:

```
# chkrootkit
```

and you will see that it detects that the loopback network interface (lo) is now in promiscuous mode. To just see this vs. all the other messages do:

```
# chkrootkit | grep PROMISC
```

If your PC is running a DHCP client daemon, you may also see that eth0 is in promiscuous mode:

```
eth0: PROMISC PACKET SNIFFER(/sbin/dhclient3[564])
```

Turn off promiscuous mode for lo:

```
# ifconfig lo -promisc
```

10 Enable System Accounting

System accounting gives us logs of all the commands that have run and terminated on the system. Let's see if we have the acct package:

```
$ which sa
```

Did "which" find the command? If not install the package:

```
# apt-get install acct
```

```
$ which sa
```

Let's run a command and see if acct records it.

```
$ whoami
```

```
# sa -u
```

Did "sa" show a record for the command?

Let's try the "lastcomm" command as well:

```
$ lastcomm sysadm
```

–End