Linux Security Ideas and Tips

Hugh Brown
Sr. Systems Administrator
ITS Enterprise Infrastructure

University of Iowa

October 8, 2014

Introduction

Who am I?

- Linux User and Sysadmin for 15+ years
- Worked in big business, small business and university environments
- Taught Linux courses for the UI

My assumptions:

- You are running Linux in a server capacity
- You are only running a handful of systems
- You aren't managing 100s of Linux systems
- The command line isn't scary
- Specifics may be Redhat–centric

Overview

- Where to get Linux
- Minimalism
- Baseline
- Patching
- Least privilege
- User management
- Remote access and ssh hardening
- Packet Filtering
- Logging
- AV protection
- Secure applications
- Further reading

Where to get Linux

- Redhat: `https://helpdesk.its.uiowa.edu/software/signin.htm`
- CentOS: `http://www.centos.org/`
- Fedora: `http://fedoraproject.org`
- SuSE: $$ `https://www.suse.com`
- openSuSE: `http://www.opensuse.org`
- Ubuntu: `http://www.ubuntu.com`
- Debian: `http://www.debian.org`

Minimalism

- Install the minimum you need, add software/packages as necessary
- Use package manager to remove any extras that you don't need
- Fewer packages/applications means less to patch later

Minimalism: Disabling unneeded services

Disable unnecessary services - Don't install them in the first place
How to determine what's listening:

```
netstat -vanp | grep LISTEN
```

```
Proto Recv-Q Send-Q Local Address   Foreign Address State       PID/Program name
tcp        0      0 127.0.0.1:5037  0.0.0.0:*       LISTEN      5464/adb
tcp        0      0 0.0.0.0:111     0.0.0.0:*       LISTEN      2017/rpcbind
tcp        0      0 0.0.0.0:22      0.0.0.0:*       LISTEN      10260/sshd
tcp        0      0 0.0.0.0:49218   0.0.0.0:*       LISTEN      9997/rpc.statd
tcp        0      0 0.0.0.0:24800   0.0.0.0:*       LISTEN      11677/synergys
tcp        0      0 0.0.0.0:40      0.0.0.0:*       LISTEN      10260/sshd
tcp        0      0 127.0.0.1:631   0.0.0.0:*       LISTEN      10073/cupsd
tcp        0      0 127.0.0.1:25    0.0.0.0:*       LISTEN      10349/master
tcp        0      0 :::111          :::*            LISTEN      2017/rpcbind
tcp        0      0 :::50162        :::*            LISTEN      9997/rpc.statd
tcp        0      0 :::22           :::*            LISTEN      10260/sshd
tcp        0      0 :::44           :::*            LISTEN      10260/sshd
tcp        0      0 ::1:631         :::*            LISTEN      10073/cupsd
tcp        0      0 ::1:25          :::*            LISTEN      10349/master
```

What/where is that program:

```
man -k <program>     ls -l /proc/<pid>/exe
```

Baseline

After install, and initial configuration: determine what "normal" is

- Which processes are running as which users (ps)?
- Which ports are they listening on (netstat/fuser/lsof)?
- Scheduled tasks (cron/at)
- Disk utilization (df)
- Log entries (logwatch, checking logs manually)
- Memory usage (top, vmstat)
- CPU usage (top, mpstat)
- Accounts on the system (/etc/passwd)
- Account access (who is "normally" logged in: w, last)

Patching

One of the foremost ways to keep your system secure is to patch regularly.

- Schedule a time to do patching (monthly or better)
- Allow for emergency patching for critical vulnerabilities
- Evaluate patches: local exploit vs. remote exploit
- Software inventory on a system (what to patch)
- Subscribe to appropriate information sources to find out when patches are available

Patching: Post Patching

Linux won't (usually) prompt you to reboot

- If you updated glibc or the kernel, you'll need to reboot
- Otherwise, restart services as needed.

  How do you find out which services need to be restarted:

```
needs-restarting # part of yum-utils package

lsof | grep inode | grep lib # RHEL5
lsof | grep DEL | grep lib   # RHEL6
```

  will output a list of all processes which have deleted files with the name `lib` in them.

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
sudo	664	root	DEL	REG	253,0	1742367		/usr/lib64/libnssutil3.so
sudo	664	root	DEL	REG	253,0	1742397		/usr/lib64/libnss3.so
firefox	3528	hbrown	DEL	REG	253,0	1742367		/usr/lib64/libnssutil3.so
firefox	3528	hbrown	DEL	REG	253,0	1742397		/usr/lib64/libnss3.so
cupsd	10073	root	DEL	REG	253,0	1742367		/usr/lib64/libnssutil3.so
cupsd	10073	root	DEL	REG	253,0	1742397		/usr/lib64/libnss3.so
sshd	10260	root	DEL	REG	253,0	1742367		/usr/lib64/libnssutil3.so
sshd	10260	root	DEL	REG	253,0	1742397		/usr/lib64/libnss3.so
qmgr	10357	postfix	DEL	REG	253,0	1742367		/usr/lib64/libnssutil3.so
qmgr	10357	postfix	DEL	REG	253,0	1742397		/usr/lib64/libnss3.so

Patching: Campus Resources

- Satellite server: Redhat products
  `https://rhnsat.uiowa.edu`
  Details:
  `http://its.uiowa.edu/campus-software-program/red-hat-linux-campus`
- Spacewalk server: Fedora and CentOS
  `http://spacewalk.its.uiowa.edu`

Least Privilege

Only give an application/user the minimally needed permissions to get the job done

- Most processes don't need to run as root, don't let them (when possible)
- Binding to a port between 1 and 1024 is root only
- Don't give out the root password
- Don't ever do `chmod 777` (unless you really mean it)
- Evaluate Setuid programs and determine if any user will need to run them
- Limit sudo privileges to just the needed commands

Least Privilege: Setuid

The setuid permission coupled with root ownership allows any user to run that command with root privileges.

Finding setuid programs:

```
/usr/bin/find / -user root -perm -4000 -print
```

Remove:

```
chmod u-s /path/to/binary
```

Least Privilege: Setuid Valid Uses

Potentially valid uses for setuid programs

- Reading files that would otherwise be protected
  /bin/sudo needs to read /etc/sudoers to validate commands
- Writing files to locations that are protected
  /usr/bin/crontab stores a user's crontab in /var/spool/cron
- Creating a raw socket (ping/traceroute)

Least Privilege: Sudo

Sudo can be used to give selective permission for performing a limited number of tasks as root

- Using `hbrown ALL=(ALL) ALL` in sudoers is full root access, be selective.
- Allowing `vi /my/config` via sudo is full root access. Allow `sudoedit /my/config` instead.
- Using sudo for select functions is a good thing.

Permissions

Tips for troubleshooting permissions:

- Check to see if SELinux is enabled: `getenforce`
- `cat /proc/<pid>/status` and look for Uid/Gid to see which user a process is running as
- Check from / down to wherever it is the app is trying to write instead of the other way round
- Use `strace` to do a system trace

```
strace -fvvvto output.strace -s 2048 -u <username> <cmd>
```

User Management

- Avoid shared/service accounts with passwords that everyone logs in with
  - Use sudo access to a service account when needed
- Use directory based user accounts (for builtin account expiration/password rules)
- Collect logs of user account activity on a different system (wtmp, /var/log/secure)
- Remove `.ssh/authorized_keys` for users that are gone/disabled.

User Management: Kerberos Authentication against AD

- Authorization against AD is an on–going project.
  For now, you'll need to populate the /etc/passwd file with an entry for each user
- Authentication is possible now

  `http://its.uiowa.edu/support/article/100409`

Remote access

ssh is a wonderful tool, hackers think so too. Ideas for securing it:

- Edit your `/etc/ssh/sshd_config`
  - Disable root login (`PermitRootLogin no`)
  - Disable Password authentication and only use keys (`PasswordAuthentication no`)
  - Leave X11Forwarding turned off (`X11Forwarding no`)
- Edit your `/etc/ssh/ssh_config`
  - Enable hashed `known_hosts files` (`HashKnownHosts yes`)
  - Disable X11Trusted (`ForwardX11Trusted no`)
- Run it on a different port than 22
- Use iptables to rate limit access and only allow trusted IPs/nets.

Packet Filtering

An easy way to keep the bad actors out is to refuse to respond

- Use host based firewalls - iptables/ip6tables
- Use TCP Wrappers

Scan your system regularly to make sure those filters are working.

Packet Filtering: Host based firewalls

Use a local host based firewall

- Make sure you have ip6tables configured if you are using IPv6
- Remember that iptables uses a first match paradigm
- Make sure you have a line that rejects/drops traffic at the end
- Test your rules with nmap from both on and off campus

Packet Filtering: TCP Wrappers

TCP Wrappers is a classic lightweight application firewall. Rules are in `/etc/hosts.allow` and `/etc/hosts.deny`

- Make use of it to fine tune what you protect
- It's a second layer of defense behind your host based firewall (ip*tables)
- Use `ldd <path/to/binary> | grep libwrap` to see if your application supports it
- The daemon name for the `hosts.allow` file is usually the name of the binary. Check documentation to be sure.

Packet Filtering: Scanning

- Scan your systems regularly to make sure the view from the outside is "normal"
- Request a scan from the Security Office
  `http://itsecurity.uiowa.edu/scan/networkscan-form.shtml`
- Use `nmap` judiciously against your own systems

Logging

There's lots of useful information to be had in `/var/log`. Attackers know that and clean it out to hide what they've done

- Use a central logging host to gather that information
- Use some sort of log analysis suite (or logwatch) to help you wade through it
- Use logging to help you determine "normal"

AV Protection

AV? We don't need no AV on Linux!

- Sometimes your Linux server is serving up files to MS Windows hosts.
- Running AV on those Linux systems is a must
- ClamAV `http://www.clamav.net`

Secure Your Applications

Look for best practice guides for any application you deploy

An Example: Apache Host Access to protect phpMyAdmin

```
<Directory "/usr/share/phpMyAdmin">
        Order deny,allow
        Deny from all
        Allow from uiowa.edu
        Allow from 128.255.
</Directory>
```

Secure Your Applications: Apache Allow/Deny Ordering

`http://httpd.apache.org/docs/2.2/mod/mod_authz_host.html#order`

Match	Allow,Deny result
Match Allow only	Request allowed
Match Deny only	Request denied
No match	Default to second directive: Denied
Match both Allow & Deny	Final match controls: Denied

Match	Deny,Allow result
Match Allow only	Request allowed
Match Deny only	Request denied
No match	Default to second directive: Allowed
Match both Allow & Deny	Final match controls: Allowed

Further reading

UNIX and Linux System Administration Handbook
`http://www.admin.com`

The Linux Command Line
`http://linuxcommand.org/tlcl.php`