Locking Down Mac OS X

Def Con 11

Jay Beale

Bastille Linux

JJB Security Consulting, LLC.

GWU Cyber Security Policy & Research Institute

Talk Contents

This talk with cover

- Auditing the core operating system.
- Choosing tigher settings.
- Automating this process with Bastille Linux.

Auditing OS X

One hour of lock-down:

- Boot security audit
- Daemon Audit
- Network daemons

After that?

- Set-UID/Set-GID Audit
- Audit cron jobs
- Configure daemons for better security

Mac OS X Roots

- FreeBSD
- Next (Steve Jobs)
- Mach (Darwin) kernel

Boot Security - Single User Mode?

Attack:

Hold down •-S during boot to boot directly into single user mode.

Defense:

- Alter rc.boot to require some authentication before allowing boot to continue.

Alter rc.boot?

secureit inserts the following into rc.boot:

```
if [ "${BootType}" = "singleuser" ];then
   password.pl
fi
```

password.pl is his perl script to check against a pre-set password. Can we break this?

(http://www.osxscripts.com/secureit.html)

Replace the kernel?

You could replace the kernel with one where single user mode is inaccessible, as described here:

http://www.securemac.com/ disablemacosxsingleboot.php

This could be a problem during your first hardware failure, though.

Boot Security - Boot from CD?

- Like most operating systems, obtaining root from boot is insanely easy.
- Hold down the Option key while booting, insert the OS X cd, and run the password utility.

Boot Security - Countermeasure

Activate the Open firmware password.

1. Boot into firmware: Command+Option+O+F
2. Type “password”
3. Type “setenv security-mode command”
4. Type “reset-all”

5. (command = password for non-standard boot)
6. (full = password for any boot)

(http://www.securemac.com/openfirmwarepasswordprotection.php)

Autologin

Prob: The login screen is bypassed, by default.

Soln: Deactivate Autologin.

(System Preferences ->Accounts->Users->
”Log in automatically as user”)

/Library/Preferences/com.apple.loginwindow.plist

Remove key autoLoginUser manually or via *defaults* command.

Restart and shutdown

Prob: Login screen allows reboot or shutdown without authentication.

Soln: Deactivate these too!

(System Preferences ->Accounts->

Login options->”Hide the restart and shutdown buttons”)

General Programs/ Network Daemons

/etc/rc runs /etc/rc.common to do some common startup that we (mostly) can't configure and to source /etc/hostconfig.

/etc/rc then runs SystemStarter, Darwin's replacement for the BSD and SysV init scripts.

SystemStarter

SystemStarter examines everything in:

/System/Library/StartupItems/foo/foo
AND
/Library/StartupItems/foo/foo

And runs them in an order it determines dynamically!

SystemStarter (cont)

Each of these startup scripts:

/System/Library/StartupItems
/Apache/Apache

Has meta-information in:

/System/Library/StartupItems/Apache
/StartupParameters.plist

Starting and Stopping Scripts

/System/Library/StartupItems/foo/foo

…takes arguments start, stop and restart.

You can shut down the current instance by running these or by running:

SystemStarter stop foo

Script Ordering

/System/Library/StartupItems/Apache/
/StartupParameters.plist looks like:

```
{
  Description     = "Apache web server";
  Provides        = ("Web Server");
  Requires        = ("DirectoryServices");
  Uses            = ("Disks", "NFS", "Network Time");
  OrderPreference = "None";
}
```

Deactivating System Daemons

/etc/rc.common sources /etc/hostconfig before running SystemStarter.

So do most of the SystemStarter scripts!

We deactivate daemons either via this file or by hacking on the scripts ourselves.

Deactivating automount

If we read

/System/Library/StartupItems/NFS/NFS

We find that the *automount* program runs by default, but that we can deactivate it by setting AUTOMOUNT=-NO- in /etc/hostconfig.

Deactivating NFS servers?

The script /System/Library/StartupItems/NFS/NFS

starts nfsd and mountd if exports exist.

It checks both /etc/exports and the output of “ nidump exports . “

Deactivating nfsiod

The script /System/Library/StartupItems/NFS/NFS

also starts the nfsiod daemon whether we like it or not. We can edit the script to deactivate this, if this box isn't an NFS client.

Deactivating the rest…

To do a thorough audit, you'd read through all the start scripts.

For the time-challenged, we'll just look at the programs on the system that are running now and find their launch scripts/variables, so we can deactivate them.

/etc/hostconfig

AFPSERVER=-NO-
APPLETALK=-NO-
AUTHSERVER=-NO-
AUTOMOUNT=-YES-
CONFIGSERVER=-NO-
CUPS=-YES-
IPFORWARDING=-NO-
IPV6=-YES-
MAILSERVER=-NO-
NETBOOTSERVER=-NO-
NETINFOSERVER=-AUTOMATIC-

/etc/hostconfig (cont)

NISDOMAIN=-NO-

RPCSERVER=-AUTOMATIC-

TIMESYNC=-YES-

QTSSERVER=-NO-

SSHSERVER=-NO-

WEBSERVER=-NO-

SMBSERVER=-NO-

DNSSERVER=-NO-

APPLETALK_HOSTNAME=*4a6179204265616c65d57
320436f6d7075746572*

Changing settings in /etc/hostconfig

We already know that we can safely deactivate automount by setting AUTOMOUNT=-NO- in /etc/hostconfig.

What about CUPS, NETINFOSERVER, RPCSERVER. and TIMESYNC?

Changing settings in /etc/hostconfig(2)

CUPS controls whether

/System/Library/StartupItems/Printing Services/PrintingServices

runs *cupsd* or not. It's safe to reset. Then run

/System/Library/StartupItems/PrintingServices/ PrintingServices stop

Changing settings in /etc/hostconfig(3)

NETINFOSERVER controls whether /System/Library/StartupItems/PrintingServices/PrintingServices

runs *nibindd* (YES) or *netinfod* (NO) or not. When set to AUTOMATIC, it starts *nibindd* if this machine is part of a non-local NetInfo domain and *netinfod* otherwise.

Changing settings in /etc/hostconfig(4)

TIMESYNC controls whether

/System/Library/StartupItems/NetworkTime/ NetworkTime

runs *ntpdate* and *ntdp* or not. It's safe to deactivate -- your call. Time sync is useful, though NTPd might be less safe. If you deactivate, run:

/System/Library/StartupItems/NetworkTime/ NetworkTime stop

Seeing what's left?

We can see what's left by running ps and learning about the processes remaining.

The first one that catches my eye is inetd.

Looking in …/StartupItems/IPServices/IPServices:

inetd?

…/StartupItems/IPServices/IPServices:

```
StartService ()
{
    ##
    # Internet super-server.
    ##
    ConsoleMessage "Starting internet services"
    inetd
    xinetd -pidfile /var/run/xinetd.pid
```

inetd and xinetd?

You can run both of these together, so long as they don’t listen on any ports in common.

Apple ships both of these listening to no ports by default.

Xinetd gracefully exits, while inetd does not.

We could modify the script, wrapping inetd in:

```
if [ `egrep -v '^#' /etc/inetd.conf | wc -l` -ge 0 ] ;
    then inetd ; fi
```

/etc/xinetd.d/ftp

```
service ftp
{
        disable         = yes
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/libexec/ftpd
        server_args     = -l
        groups          = yes
        flags           = REUSE
}
```

What programs are left? (1/2)

/System/Library/CoreServices/…

/sbin/autodiskmount

/sbin/init

/sbin/mach_init

/usr/libexec/crashreporterd

/usr/sbin/blued

/usr/sbin/lookupd

configd

What programs are left? (2/2)

cron
dynamic_pager
kextd
netinfod
syslogd
update
/usr/sbin/mDNSResponder
DirectoryService

autodiskmount

autodiskmount is used for mounting removable media and OS X .dmg (disk image) files.

You can deactivate it by commenting it out of (or deleting):

/System/Libraries/StartupItems/

Disks/Disks/

mDNSResponder

mDNSResponder is the core registration daemon for Rendevous. Rendevous is Mac's broadcast/discovery system.

You can deactivate it by commenting it out of (or deleting):

/System/Libraries/StartupItems/
mDNSResponder/mDNSResponder/

Deactivating the rest…

To do a thorough audit, you'd read through all the start scripts.

This is covered in detail at:

www.bastille-linux.org/jay/killing-osx-daemons.html

More to do?

Our next step at this point should be to see what's still listening on the network and make sure that it makes sense.

```
# netstat -anp tcp
# netstat -anp udp
# lsof -i  tcp:port
# lsof -i udp:port
```

TCP Audit

```
# netstat -anp tcp | grep LISTEN
tcp4    0      0  127.0.0.1.1033         *.*    LISTEN

# lsof -i tcp:1033
COMMAND    PID USER   FD   TYPE     DEVICE
      SIZE/OFF NODE NAME
netinfod   277 root    6u  inet 0x01c92d1c      0t0  TCP
      localhost:1033 (LISTEN)
netinfod   277 root    7u  inet 0x01c9123c      0t0  TCP
      localhost:1033->localhost:963 (ESTABLISHED)
```

netinfod?

Netinfod is the local-only Netinfo daemon used by the operating system for local lookups.

- It only listens on loopback.
- It’s not clear whether OS X will run without it.

UDP Audit - Netstat

```
# netstat -anp udp
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign
      Address        (state)
udp4       0      0  *.49231                *.*
udp4       0      0  *.49227                *.*
udp4       0      0  127.0.0.1.1033         *.*
udp4       0      0  *.514                  *.*
udp4       0      0  *.68                   *.*
```

UDP Audit - Port 49231 - lookupd

```
# lsof -i udp:49231
COMMAND   PID USER   FD   TYPE     DEVICE
      SIZE/OFF NODE NAME
lookupd 2772 root    8u  inet 0x01ae6cb0      0t0  UDP
      *:49231
```

Lookupd is “an information broken and cache” for most information about the system. It consults Netinfo, NIS, DNS, and even the files in /etc/. We should probably leave it alone.

UDP Audit - Port 49231 - lookupd

```
# lsof -i udp:49227
COMMAND  PID USER   FD   TYPE     DEVICE SIZE/OFF NODE
      NAME
lookupd 2772 root    4u  inet 0x023c5d80      0t0  UDP *:49227
```

This is also lookupd. Let’s leave this alone.

UDP Audit - Port 1033 - netinfod

```
# lsof -i udp:1033
COMMAND  PID USER   FD   TYPE     DEVICE
      SIZE/OFF NODE NAME
netinfod 277 root    5u  inet 0x01ae6be0      0t0  UDP
      localhost:1033
```

This is also netinfod. Remember, this is only listening on loopback.

Let's leave this alone.

UDP Audit - Port 514 - syslogd

```
# lsof -i udp:514
COMMAND PID USER   FD   TYPE     DEVICE
      SIZE/OFF NODE NAME
syslogd 253 root    4u  inet 0x01ae6e50      0t0  UDP
      *:syslog
```

This is syslogd, though it shouldn't be listening on the network unless it's the central syslog server.

Following up on syslogd

While syslogd has bound to the syslog port (514), it doesn't appear to accept syslog messages from other hosts.

This is consistent with the man page for syslogd:

> -u Select the historical ``insecure'' mode, in which syslogd will accept input from the UDP port. Some software wants this, but you can be subjected to a variety of attacks over the network, including attackers remotely filling logs.

One wonders whether it will accept spoofed messages pretending to be from this host…

UDP Audit - Port 68 - Configd

```
# lsof -i udp:68
COMMAND PID USER   FD   TYPE     DEVICE
        SIZE/OFF NODE NAME
configd 110 root    7u  inet 0x01ae6f20      0t0  UDP
        *:bootpc
```

Configd provides the DHCP client on OS X. We'll need to keep it on for this system.

Next Steps

- Do a Set-UID/Set-GID audit
- Audit cron jobs
- Audit daemon configurations
- Do a Permissions audit

Set-UID audit/Set-GID audit

#find / -type f -perm -04001 -ls >suid-files

#find / -type f -perm -02001 -ls >sgid-files

#find / -type f -perm -04001 -user 0 -ls \ >suid-root

#find / -type f -perm -02001 -group 0 -ls \ >sgid-root

#find / -type f -perm -02001 -group 80 -ls \ >sgid-admin

www.bastille-linux.org/jay/suid-audit.html

Cron Jobs

On OS X, we look at /etc/crontab:

```
# Run daily/weekly/monthly jobs.
15      3       *       *       *    root    periodic daily
30      4       *       *       6    root    periodic weekly
30  5       1       *       *    root    periodic monthly
```

31 This just runs the periodic program.
We need to audit /etc/periodic/*.

Cron audit - /etc/periodic/*

```
# ls /etc/periodic/*
/etc/periodic/daily:
100.clean-logs  500.daily

/etc/periodic/monthly:
500.monthly

/etc/periodic/weekly:
500.weekly
```

Daemon Configurations

When you harden a system, you generally spend part of your time deactivating daemons, but also good deal of your time changing the configurations of the remaining daemons to something stronger.

Some of this involves running things as non-root users and creating jails/chroot prisons.

The rest involves tweaking config files.

Permissions Audits

It's very interesting how some weak permissions around the system can give you far more access than you should have.

Consider this question:

What could a user on your system do if he realized that a commonly-used binary was in a world-writable directory?

Permissions Audits

Begin your permissions audit with a search for world-writable files and directories then!

```
# find / -type f -perm -02 -ls >world-writ-files
# find / -type d -perm -02 -ls >world-writ-dirs
```

Bastille Linux

Bastille Linux is a hardening script for five Linux distributions, HP-UX and Mac OS X.

It can automate much of what we're doing here.

www.bastille-linux.org

Read my articles for more…

My articles have focused on Linux lockdown.

You can these, along with lockdown articles on Mac OS X, on:

www.bastille-linux.org/jay/