HARDENING GUIDE

REDHAT LINUX

Prepared By Philip Do

1 REDHAT LINUX INTRODUCTION

1.1 Overview

The purpose of this document is to provide guidelines for hardening a host running Redhat Linux. While most issues will apply to Linux (and other Unix's) in general, examples will be in terms of Redhat Linux.

2 PRE-INSTALLATION

Before the installation even begins, there are some important factors to consider.

2.1 Security is a Process

Hardening a newly built (or existing) host does not inherently protect it against compromise. The security of a host depends on the security of the network, of the hosts adjacent to it, and, most importantly, upon the vigilance of it's administrator.

Failure to continually monitor and maintain a host will put it as much risk (if not more) than failing to make the configuration changes suggested in this document.

The compromises discussed in 2.2 Security versus Ease of Use versus Cost, and the vast complexity of modern systems ensure that ***no*** system can be totally secure. It is important that, if the worst happens, an incident response plan is in place to guide staff as to what actions they should take.

Security is a process, a *life cycle*, not an event, or an action. Host hardening is just the first step.

2.2 Security versus Ease of Use versus Cost

Host security, cost and it's relative ease of use (and maintenance) are all inversely proportional to each other. The systems administrator must carefully weigh these factors when planning any system, with input from the company's Security Policy and UNIX Security Baseline.

2.3 Plan the Install

Before installation media is first loaded, the installation should already have been planned. The following should all be known:

- The purpose of the host
- What services the host will provide
- Where in the network it will be located
- Network configuration
- Disk and file system layout

All of these factors effect the decisions that must be made as the host is installed and hardened. If they are not defined until afterwards, then there is a risk that critical elements may be missed.

3 INITIAL BUILD

3.1 Environment

The initial installation and hardening should occur with the host either totally disconnected from the network (ideal, but not practical), or on an isolated, protected segment. The host is at most risk of compromise before it is properly hardened, so care must be taken to protect it.

3.2 Keep Records

Keep a record as you build the host. Not only will this aid you in debugging problems later, it allows future hosts to be built in a standard fashion.

Note down the software packages installed, how they were installed (from source or RPM) and the exact version number. It is much easier to build such a list as a host is built, rather than trying to assemble it afterwards. Whenever an applicable security advisory is released, systems support personnel can compare the vulnerable versions of the software against the list of software installed within the company and quickly determine which, if any, hosts are affected.

3.3 Disk/File System Layout

The disk and file system layout of a host should be made such that different partitions can be assigned different security related parameters when mounted. These parameters should be configured in `/etc/vfstab` and apply to the Linux 'ext2fs' file system.

Table 1 - File System Parameters

Parameter	Description
nosetuid	Disables processing of the 'setuid'[1] and 'setgid'[2] flags within the partition.
Ro	Prevents any form of writing to the partition. (Read Only)
Noexec	Prevents programs on the partition being executed.

All partitions containing static system related binaries and libraries should be mounted read-only.

All partitions *not* containing static system related binaries should be mounted with 'setuid' and 'setgid' disabled.

[1] **setuid**: When set on an executable, this flag causes it inherit the user permissions of its *owner* rather than those of the user executing it. It is particularly dangerous on root owned files, though it is (safely) used on a number of system binaries.

[2] **setgid**: Similar to the setuid flag, the setgid flag causes a program to inherit the group permissions of it's owner rather those of the user executing it.

All partitions containing only stored information should have file execution disabled.

Suggested flags for common partitions are suggested below.

Table 2 -Suggest File System Parameters

Partition	Flags	Contents
/	Ro	Root partition
/usr	Ro	Core system programs and supporting resources.
/etc	Ro	System configuration information
/var	noexec , nosetuid	'Variable' files. Temporary state information, logs, etc.
/home	Nosetuid	User home directories.
/tmp	noexec, nosetuid	Temporary files.
/opt	ro, nosetuid	Optional packages
/root	---	Root home directory
/lib	ro, noexec, nosetuid	System libraries.
/sbin	Ro	System Administrator's binaries
/bin	Ro	System binaries

WARNING: Setting the read-only flag on /etc will require that the system be rebooted to single user mode in order to make any system configuration changes. Likewise, setting read-only for /usr will require downtime when new packages are installed.

WARNING: Packages that operate in an unusual way may be crippled by the use of these flags. Be careful.

3.4 Package Installation

Perform a custom installation and only select the packages that are needed for the host to perform its duties. Exactly which packages to select depends very much on the intended purpose of the host.

Once the host is booting, go back over the RPM[3] database and remove any that are not needed.

3.5 Vendor Recommended Patches

At this stage, check the Redhat WWW site (http://www.redhat.com/) for any vendor recommended security patches and updates. All patches and update that are applicable should be installed.

[3] **RPM:** Redhat Package Manager: A system used to distribute entire software packages under Linux. It also allows for easy package removal, listing, conflict testing, and verification.

4 POST-INSTALLATION CONFIGURATION

4.1 Remove Redundant User Accounts

The default Redhat build ships with a number of users defined that are unnecessary.

User accounts such as ‘uucp’, ‘mail’, ‘new’, ‘operator’, ‘games’, ‘gopher’, etc. can safely be removed.

WARNING: Editing the `/etc/passwd` and `/etc/shadow` can render your system unusable! Modification of these key system files should be made using the `vipw` editor or the `userdel`/`userdel`/`usermod` commands.

4.2 Disable All Unnecessary Daemons

By default, Redhat Linux enables a large number of daemons that are not needed, some of which are a security risk on an Internet connected host.

The files in the subdirectories of `/etc/rc.d` control what daemons are started at boot time. They should be examined carefully, and any that are not needed for the secure operation of the host should be disabled.

The `/etc/rc.d/init.d` directory contains the actual scripts that control the start-up of the daemons. For it to be started at boot time, however, it must be called as described next.

The start-up and shutdown of system daemons is controlled by the location and name of a *symbolic link* to the script itself in `/etc/rc/init.d`.

Scripts located in `/etc/rc.d/rc0.d/` are called when the system moves to run level 0. Scripts located in /etc/rc.d/rc1.d/ are called when the system moves to run level 1, etc.

Within these directories, the links are formatted as follows:

```
(S|K)##<comment>
```

That is, they start with either an ‘S’ or ‘K’, followed by a number, followed by a string of text.

The trailing text is discarded by the system and is commonly used to tag it with a name a human systems administrator will understand.

The initial character defines whether it is a start-up or shutdown script. Scripts starting with a ‘S’ are called with the parameter ‘start’. Scripts starting with a ‘K’ are called with the parameter ‘stop’.

The number is an ordering control, with files being called in ascending numeric order.

To disable a script, it must only be renamed, not deleted. A common technique is to replace the initial ‘S’ or ‘K’ with the lower case equivalent.

4.3 Disable Unnecessary Services

The Internet Superserver, otherwise known as the inetd, controls most IP services. The actions of this daemon are controlled by the `/etc/inetd.conf` file, which, by default, enables many unneeded services.

In general, the “small services” (echo, discard, daytime, chargen, time) and “information services” (netstat, systat, finger, auth, identd) should all be disabled.

Login services (telnet, ftp, shell, login, rlogin, etc.) should be disabled and replaced with SSH.

Unless some specific services are required from the inetd (such as a public FTP server), it may well be possible to disable all of the inetd services and disable the inetd itself. (Using the procedure discussed in section 4.2).

4.4 Secure LILO

The Linux Loader (lilo) needs to be secured to prevent intruders with local access to the host compromising the host from the console.

The ‘protected’ keyword should be used in `lilo.conf` in conjunction with a password to ensure that no parameters can be passed to lilo (such as booting the system in single user mode) without entering the password.

The `lilo.conf` file should be made immutable using the following command:

```
chattr +i lilo.conf
```

This file should also have permissions 0400.

4.5 Remove Empty Crontabs

All empty cron tables in `/var/spool/cron/crontabs` should be removed.

4.6 Setup Logging

The system logger daemon (syslogd) should be configured to log useful messages both locally and to a log server. On a modern system, in conjunction with log rotation, it should be possible to log all useful information without disk space concerns.

Authentication and other security related information should be logged to a separate file from the main system log.

A simple configuration (removing priority handling for critical events) is as follows:

```
*.info;mail.none     /var/log/messages
auth.info            /var/log/authlog
local3.info          /var/log/tcpdlog    # for TCP
Wrappers
```

This should be entered into `/etc/syslog.conf`.

Replacements or enhancements to the system logger (such as Nsyslog) should be considered, as should automated log monitoring software, though the latter is best implemented on a central network logging server.

4.7 Control Access

All remote access to the host should be made using the Secure Shell (SSH). SSHv2 should be used where possible, with public key authentication. All other methods for remote access should be disabled. (See section 4.3).

4.7.1 Source Based Access Control

Source based access control should be implemented to restrict exactly where users can login from. This should be done using the /etc/hosts.allow and /etc/hosts.deny files.

WARNING: Under no circumstances should a `/etc/hosts.equiv` file be used!

4.7.2 Banners

All incoming logins should be presented with a login banner, ideally before logging into the host. This banner should notify the user that unauthorised use is prohibited and that monitoring will take place. A suitable banner is shown below.

```
WARNING
This system is for the use of authorized users only.
Individuals using this computer system without
authority, or in excess of their authority, are subject
to having all of their activities on this system
monitored and recorded by system personnel.

In the course of monitoring individuals improperly using
this system, or in the course of system maintenance, the
activities of authorized users may also be monitored.

Anyone using this system expressly consents to such
monitoring and is advised that if such monitoring
reveals possible evidence of criminal activity, system
personnel may provide the evidence of such monitoring to
law enforcement officials.
```

4.7.3 Superuser Access

Access to the root (superuser) account must be *strictly* controlled. It should be impossible to login as the root user remotely. (Ensure that only the "physical" terminals are listed in `/etc/securetty`, and that '`PermitRootLogin`' is set to '`no`' in the SSHd configuration).

Users requiring true root access should be given the root password, and added to a group such that they can utilise the '`su`' command.

Users requiring some aspect of root authority, but not requiring full root access, should be controlled using '`sudo`'. Care should be taken to configure this in a granular manner. Access should be reviewed frequently and revoked if no longer required.

4.8 Network Related Changes

Ensure that the host is not functioning as a router by disabled IP forwarding using the following command:

```
echo 0 > /proc/sys/net/ipv4/ip_forward
```

Likewise, enable SYN cookies using the following command:

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

4.9 Use Network Protection

As part of a secure network design (refer to [1]), packet and connection filtering utilising products such as IP Chains and TCP Wrappers should be implemented.

A minimal rule set should be used to protect the host, strictly restricting traffic to what is required for the host to perform it's duties.

4.10 Use Time Synchronisation

Time synchronisation should be implemented to ensure that all system clocks in a trust domain are synchronised to each other. This is important to for maintenance of some time dependant cryptographic protocols, but is more important for effective correlation of events across multiple hosts.

4.11 Remote Compilers and Interpreters

All compilers and interpreters should be removed from a system unless they are required for a specific (important) application on a host. (For example, many applications require PERL to be able to function).

Removing these files makes many automated attacks against a host much more difficult, as the intruder must first load and build their own compiler or interpreter before being able to utilise their tools.

4.12 Implement a Backup Scheme

Variable information should be backed up on a regular basis to ensure that the host can be recovered quickly in the event of a failure. At the very least, a 'day 0' backup (as described in section 6.1) should be taken.

Some hosts, such as DNS servers, contain very little mutable information, and backups can be achieved by simply copying key files to a backup server every so often.

Other hosts, such as file servers, mail servers, database servers, require a more thorough and capable backup solution.

4.13 Check File Permissions

The following sections discuss a variety of configuration changes relating to file permissions.

4.13.1 Logs

All log files in should have the Append-Only flag set to help retain the integrity of log information if the system is compromised. Note

that this will only slow down an intruder, but it also provides benefits against accidental modification.

```
chattr +a /var/log/messages
```

The log rotation script will need to reset this flag, and should make historic logs immutable.

World read (and write!) permissions should be disabled on all log files, and indeed, the log directory itself.

4.13.2 Verify setuid and setgid

All files with the setuid and setgid bits set should be located and verified. Any not required should be disabled.

A list of all files with these bits set can be retrieved using the following command.

```
find / -type f \( -perm -04000 -o -perm -02000 \)
```

4.13.3 Check for World Writable Files

There should be no files with the world writable flag set. Check for this using the following command.

```
find / -perm -2 ! -type l -ls
```

4.13.4 Ensure All Files Have an Owner

All files on the system should have a valid owner and group. Files with UIDs or GIDs without a corresponding user or group can cause vulnerabilities to arise when users or groups accidentally inherit them later.

Extracting archives while logged in as root[4], or by deleting users and not cleaning up their files generally causes files of this sort.

These can be located using the following command:

```
find / \( -nogroup –o –nouser \) –ls
```

4.13.5 Configuration Files

In general, no files in the `/etc` directory should be group writable. Consider setting the immutable flag on all critical configuration files.

[4] This is because 'tar' retains ownership information when it creates an archive. When a file is extracted as root, files are created with their original ownership information, which may be completely bogus if it was created on a different host to the one it is extracted on.

5 APPLICATION SPECIFIC CONSIDERATIONS

5.1 Be Wary

Always be wary of any application that must be installed on a host. Be especially wary of NIS, NFS and SMB, RPC and X11. These should not be necessary on an Internet facing server, and introduce a significant risk if they are used.

Ensure that any package installed has been acquired from a trusted source and that it's signature is valid.

5.2 Replace sendmail

Many UNIX systems report status using the mail mechanism, and as such it cannot be totally disabled. The system should not accept incoming SMTP mail, however. (Unless it is designed as a mail server, of course).

The default mail delivery agent for Redhat Linux is the venerable sendmail. While adequate, its poor security track record and complexity of configuration make it a poor choice. Products such as qMail and PostFix are a preferable replacement.

5.3 Avoid UID 0

Few applications need to run as root, and most do only because the time has not been taken to install them to run as a different user.

As a rule, only applications that bind a reserved port (one that is less than 1024) or perform low-level system functions need to run as root. (It is possible for some applications binding reversed ports to run as a non-root UID).

5.4 Imprison Where Possible

Consideration should be made to utilising a chroot "jail" for any network server application. Because these servers accept incoming requests from the network, they are at the most risk of attack. If the application is in a chroot jail, then the amount of damage that can be caused is minimal.

NOTE: Utilising chroot for a process running is root gains nothing as a user with root permissions can break a chroot "jail".

Implementing chroot can be a very complicated and time-consuming task for even a moderately complicated application. Reference

should be made to the Security Policy to determine if such expenditure is necessary.

5.5 Enhanced Security with SSL

Some applications and protocols can be enhanced by implementing them using SSL. This allows the client to verify the identity of the server and for authentication information to be exchanged in a secure fashion, defeating man-in-the-middle and network sniffing attacks.

The best examples of this are various POP and IMAP mail servers in conjunction with SSL enabled clients (such as MS Outlook, SSL Fetchmail, Netscape, etc.)

6 LAST STEPS

6.1 Implement an Integrity Checker

A host integrity checker (such as Tripwire, Aide, Osiris, Enterprise Security Manager, etc.) should be implemented to monitor the host for changes in key configuration files, system binaries and libraries, etc.

Before the host is moved into production, a first snapshot of the host should be taken and committed to read only media so it can be verified against the running configuration regularly.

In addition, a 'day 0' backup may prove useful.

6.2 Audit the Host

Take this opportunity to verify everything that is running on the machine is what is expected. Verify the process listing after reboot. Scan the machine for vulnerabilities (SATAN/SAINT, Nessus, ISS, etc.) and open ports (NMAP), and test the configuration of all network protection systems.

This information should be recorded and added to a host specific baseline and checked for variance regularly

6.3 Backup the RPM Database

In addition to the Integrity Checker baseline (see section 6.1), a backup should be taken of the RPM installation database. This provides an extra level over the Integrity Checker, and is especially important if the Integrity Checker does not support Linux easily.

The two files to backup are `/var/lib/rpm/fileindex.rpm` and `/var/lib/rpm/packages.rpm`. These can then be verified against the current RPMs installed using the following command.

```
rpm -Va
```

6.4 Build an Incident Disk

It is always best to prepare for the worst, so the opportunity should be taken to prepare an Incident Disk for the specific platform the host is using. This disk should contain copies of key forensics tools such as `ls`, `ps`, `find`, `lsof`, `md5`, `strings`, `strace`, `ltrace`, `nm`, `ldd`, `netstat`, `ifconfig`, `top`, `du`, `kill`, etc.

These tools should all be statically linked if they are not already.

This disk should then be made read-only, backed up, and stored safely.

Referecencs

1. John H. Terstra, Paul Love, Ron Rech, Geoff siver, Tim Silver, Tim Scanlon, Mike Sherman. Linux Hardening: Bulletproof your system before you are hacked. Osborne, 2004.
2. Wesley, J Nooman. Hardening Network Infrastructure. Osborne, 2004.
3. Eleen Frisch, Guru Guidance Hardening Linux Systems, guru.html.
4. Aeleen Frisch. Essential System Administration. O'Reilly & Associates, Inc 2002.
5. Evi Nemeth, Garth Snyder, Scott Seebass, Trent R. Hein. Unix System Administration Handbook. Prentice Hall.
6. Taylor Merry. Linux Kernel Hardening. SANS Institute 2003.
7. Wreski D. Next Generation of Kernel Security. Security.html, 2003
8. Smalley, S., Fraser, T., Vance., C.. Linux Security Modules: General Hooks for Linux.
9. Dragovic, B. Linux Security Protection System. Security.html.
10. Hallyn, S., Kearns., P. Domain and Tyepe Enforcement for Linux. Paper.html.