# Securing RedHat Linux

*How to implement RedHat Linux in a secure manner*

## Allaire Security White Papers Series

(Version 1.0)

## Abstract

| | |
|---|---|
| **Title** | Securing RedHat Linux |
| **Date** | January 8, 2001 |
| **Product** | RedHat Linux |
| **Target Audience** | Linux Administrators |
| **Abstract** | Securing a Linux server can be a difficult process, considering the vast number of security advisories an administrator must keep track of. This document and other lockdown documents are Allaire's effort toward making this job a little easier. |


Allaire Corporation • One Riverside Center • 275 Grove Street • Newton • MA • 02466
www.allaire.com • info@allaire.com • (617) 219-2000 •
security issues: secure@allaire.com
document feedback: lockdown@neohapsis.com

# Table of Contents


Abstract ..... 2
Table of Contents ..... 3
About this document ..... 4
Installation considerations ..... 4
    Package selection ..... 4
    Individual package installation ..... 6
Installing patches and updates ..... 6
Unneeded services ..... 7
    Startup services ..... 7
    Inetd services ..... 9
Suid applications ..... 10
Additional service configurations ..... 12
    Ftp ..... 12
Miscellaneous ..... 13
    SSH ..... 13
    NMAP ..... 13
    Locking rhosts ..... 14
    Legal banners ..... 14
    Using development tools ..... 14
    TCP_Wrappers ..... 15
    IPChains ..... 15
    PAM (Pluggable Authentication Modules) ..... 15
    OpenWall Project ..... 15
    Bastille Linux Project ..... 15

## About this document

This document is targeted at the Linux system administrator who is deploying RedHat Linux for web/Internet-based use. This document assumes the installation of the RedHat Linux version 6.1, which is the current version at the time of writing.

## Installation considerations

Starting with version 6.1, RedHat has switched to using a graphical installation-utility to select the various system configurations and options in which you may be interested. A full walk-through is unnecessary; instead we will highlight key installation points.

When prompted for password options, verify that 'Enable MD5' and 'Shadow passwords' are checked. MD5 allows for the use of a stronger password-storage-algorithm (and passwords greater than eight characters). Shadowing the password file allows the passwords to be kept in a separate, more secure file.

Always pick a strong root password. Preferably, the password should be at least eight characters long containing numbers, letters and non-alphanumeric (for example, !@#^%$) characters.

RedHat Linux systems also exhibit a bug where any user account assigned a password during installation -- including the root -- has the password stored using the older crypt() algorithm, not MD5. So, any account added during the install process must use 'passwd' to update the password after the system is fully installed, to use the stronger MD5 password-storage-algorithm.

### Package selection

When implementing a secure server, use the 'minimalist' approach: install only what you need. The less installed the fewer the potential problems. And, there will be fewer entry points for an intruder. The GUI interface is a perfect example—we recommend not installing any X-Windows components. If you look at the history of RedHat patches, you'll find a hefty amount are for various vulnerabilities found in GUI-related packages. This document assumes that command-line administration is acceptable.

When choosing the installation type, select 'Custom.' This lets you pick the packages you'll install. When selecting basic package groups, consider choosing:

- *Networked Workstation*. Provides various tools and utilities otherwise associated with network connectivity.

- *WWW Server.* Installs the Apache web server package.

- *Development.* Installs the compiler and associated development tools and libraries. Install a compiler because it lets you create new binaries using the latest source code, which may include security patches. However, *not* installing a compiler may be preferable in environments where non-production servers are deployed for procedures like compiling.

It's also important to check the 'Select Individual Packages' option. This lets you specifically choose what will be installed. Adjust the following categories:

*Applications*
- *Archiving.* Add: tar
- *File*. Add: gzip
- *Internet*. Remove: ncftp, rsh
- *Publishing*. Remove everything
- *System*. Remove: knfsd-clients, rdist, rdate

*Development*
- *Debuggers.* Remove everything
- *Languages*. Add: perl
- *Tools*. Remove: ElectricFence

*System Environment*
- *Base*. Remove: chkfontpath, piranha, yp-tools
- *Daemons*. Remove: xfree86-xfs, lpr, pidentd, rusers, rwho, routed, portmap, php, mod_perl, tftp, timed, ypbind
- *Libraries*. Remove: xfree86-libs
- *Shells*. Add: bash2

*User Interface*
- Remove all packages from all subgroups

***Note:*** *It is not necessary to select packages for installation now. Use the RPM utility (RedHat package manager described below) to install additional packages as needed. It is better to install the minimal list above and add other packages later than it is to add packages now and not use them.*

Notice an FTP server is not on the above list. RedHat comes with multiple FTP servers and by selecting the 'FTP Server' package, you install all of them. Instead, use RPM to install only the ones you want.

**Individual package installation**
To install new packages, access the package's rpm file, the original of which is shipped on the installation CDROM. Simply mount the CDROM to gain access to the RPMs by running:

```
mount /dev/cdrom /mnt/cdrom
cd /mnt/cdrom/RedHat/RPMS/
```

Each RPM is listed in the following format:

```
package_name-version-architecture.rpm
```

If you need an FTP server (most virtual hosting companies do), install it now. To install packages, use the 'rpm –i' command. In this case, you can install wu-ftpd (the FTP server) by running:

```
rpm –i wu-ftpd*.rpm
```

## Installing patches and updates

Download and install any new patches or updates now since installing a patch later may undo some of the configuration changes detailed in this document.

All patches for RedHat Linux are released on the updates.redhat.com FTP site. You'll find a list of mirror FTP sites located at http://www.redhat.com/mirrors.html. The patches are located in the directory specified by your architecture (i386, sparc, alpha) and distribution version (6.1) in the appropriate update directory (consult the mirror list for where your particular mirror stores the update directory).

Grab all patches listed on the update site by using the 'mget *' command, if your FTP client supports it.

Installing the patches is easy—you need run only the following command in the same directory that has all the previously downloaded updated RPMs:

```
rpm –F *
```

The –F option tells RPM (RedHat package manager) to update the package, but only if it's installed. This is preferable to the older –U option, which updated the package, but would also install it if it wasn't previously installed.

***Note:*** *Kernel upgrades require a reboot for the updates to take affect, whereas rebooting after glib updates is merely recommended since many system applications use the base glib libraries.*

## Unneeded services

By default, Linux installs and activates some frequently unneeded services. Most good security practitioners agree: run only the services and subsystems you actively use. By disabling unneeded or unused services, you greatly reduce your system's potential vulnerabilities.

### Startup services

The system should always start with as few services as possible. Service initialization is controlled by the scripts found in the "/etc/rc*.d" directories. We will focus on /etc/rc3.d, which is the default run-level for RedHat Linux.

In the /etc/rc3.d directory you'll see various scripts that begin with the letters 'K' or 'S.' Each script controls its service counterpart (i.e. S30syslog controls syslogd). All scripts beginning with the letter 'S' are started at this run-level, while all scripts starting with the letter 'K' are stopped (or 'K'illed). The number after the initial S or K specifies the order in which the scripts should be called.

Disabling scripts is as simple as renaming the specific 'S' entries to 'K' entries. For example, if you want to disable the Sendmail service, you would run

```
mv S80sendmail K80sendmail
```

Renaming the script is preferable to deleting it since you can re-enable the service at boot simply by restoring the original name. Keep in mind, however, that while renaming the script will keep the service from starting at boot, if the service was running beforehand it will continue to run until the system is rebooted.

RedHat now has a command called chkconfig that will handle the renaming for you. To disable Sendmail, for example, you would run

```
chkconfig sendmail off
```

You can stop services by calling the script with the 'stop' parameter before you rename it, i.e.

```
./S80sendmail stop
```

Disabling services is entirely dependent on what the system needs and for what it is being used. Disable the following, providing you're not using these services:

- *Laptop services*. If your Linux server is not on a laptop, you do not need laptop power management or PCMCIA support. Disable the following:
  - S16apmd
  - S45pcmcia

- *Atd*. A scheduled program manager that lets users and the system run commands at certain points of time. However, this functionality duplicates crond. To use only crond, disable the following:
  - S40atd

- *Gpm.* Provides console mouse support. If you do not need mouse support on the text console of the server, disable the following:
  - S85gpm

- *linuxconf*. A new tool that allows for remote administration and configuration of your Linux system. However, its remote-access protocol is not very secure and has contained vulnerabilities in the past. Use a secure access method, such as SSH (described below) to access your system and use the linuxconf command-line utility locally. To disable remote access to linuxconf, disable the following:
  - S99linuxconf

- *Sendmail*. The use of Sendmail is dependent on whether or not your host must function as a mail server (this constitutes receiving incoming email; outgoing email is possible without Sendmail). If you're not using Sendmail, disable it. Sendmail has a seemingly never-ending set of security flaws. If you disable Sendmail at boot you *will* be able to send outgoing email -- contrary to popular belief -- however, Sendmail does require an extra entry be added to cron (details below). Startup is controlled by the following:
  - S80sendmail

Other startup services/scripts include:

- *S05kudzu.* Monitors for hardware changes on boot (required).

- *S99local.* Runs /etc/rc.d/rc.local (can be disabled if no required configuration commands are contained in rc.local).

- *S50inet.* Starts inetd, which controls access to network services (required if you need a service specified in /etc/inetd.conf)

- *S30syslog.* The system log daemon (required)

- *S40crond.* Task scheduler (required)

- *S10network.* Startup and configure the network interfaces (required)

- *S20random.* Reseeds the random number generator (required)

- *S75keytable.* Loads the required keyboard map (required)

- *S25netfs.* Mounts NFS, SMB and Novell network shares into the filesystem (disable if you don't need to remotely mount network file shares on boot)

- *S85httpd.* Starts the HTTP server (required if you want to run a web server)

**Inetd services**
Many services are also started by requests from the network. Inetd, a popular Unix service, traditionally handles these requests by starting up the proper service in response. The services inetd will allow are specified within the /etc/inetd.conf configuration file. By default, RedHat Linux has many services enabled. Rather than figuring out which ones to disable, simply disable all services and then consider which ones to enable. Luckily, none of the services launched from inetd are required for core system operation (other than telnet, which is used for remote system-administration; however, telnet should be replaced with SSH).

You can disable a service by commenting out the appropriate command line in /etc/inetd.conf.

The common services launched from inetd include:

- *FTP.* The FTP server lets your system act as a fileserver for ftp-capable clients. Most ISPs provide FTP functionality to their client base letting them upload material to hosted websites. FTP transmits passwords in plaintext, which is a *serious* security concern. Whenever possible, limit your users to using a more secure method of file transfer, such as scp and disable ftp. Scp is available in most SSH distributions, which is discussed later in this document.

- *Telnet*. Telnet is a classic service that lets you log into your system remotely. Like ftp, it also transmits passwords "in the clear." But telnet access is an even greater security concern than FTP because it allows remote shell access to your system. Consider disabling telnet and replacing it with something like SSH (secure-shell). SSH is discussed later in this document.

- *Shell/login/exec* (also known as rshd, rlogind and rexecd). A suite of services that allow for remote login and the remote execution of commands. You gain the same functionality by using SSH or telnet. Disable this suite unless you have applications specifically designed to use it (and even then this is a security concern).

- *POP/IMAP.* POP (version 2 and 3) and IMAP are services for remote users to retrieve mail. Note that a user's authentication information (login name and password) travel in cleartext on the network for these services, and should therefore be considered insecure. However, an equivalent replacement has

not yet been introduced, so virtual hosting-providers that provide email services are stuck using these insecure services. Combat the insecurity by using virtual user accounts and otherwise limiting the ability for POP/IMAP users to log into the system using other services.

- *Finger/cfinger.* Allows attackers to gain user information. Fingerd and cfingerd should be disabled.

- *TFTP*. Similar to FTP, TFTP is a very simple file-transfer service. Unlike FTP, TFTP uses *no* authentication. It is the source of numerous security problems and should always be disabled. If you require the use of TFTP for router images, boot images, etc., manually enable it on a case-by-case basis.

- *Linuxconf.* Linuxconf allows remote configuration of your RedHat Linux server. This poses a security risk and should be disabled. (Note: it is enabled by default).

- Other legacy services that should be disabled:
  - *Comsat*
  - *Talk/NTalk/DTalk*
  - *UUCP*
  - *SystatNetstat*
  - *Time*
  - *Auth*

## Suid applications

The set user id bit (referred to as the 'suid' bit) allows an application to run under the context of another user. This is typically used to let normal users interact with files and system services/processes to which only root has access. These applications are usually designed with security in mind, but it's possible for a local user to take advantage of a vulnerable application and run arbitrary commands under the suid user context. Limiting the amount of suid applications on your system will greatly increase your security posture.

You can generate a list of all suid applications on your system by running

```
find / -type f –perm +6000
```

To disable a suid application run

```
chmod –s /path/to/application
```

***Note:*** *Disabling a suid application does not mean it cannot be used. Suid allows normal users to use the application when they wouldn't normally be able to.*

Removing the suid bit prevents normal users from using the application (or the functions that require extra privilege); however, the root user can continue using the applications.

The following is a list of suid applications we recommend disabling, as well as the reason they are suid (what they let users do):

- */usr/sbin/usernetctl*. Manipulate network interfaces
- */usr/bin/sperl** . Execute suid perl scripts
- */usr/bin/suidperl*. Execute suid perl scripts
- */bin/mount*. Mount filesystems, if user is in /etc/fstab
- */bin/umount*. Unmount filesystems, if user is in /etc/fstab
- */usr/bin/chage* View how many days before a user's password expires
- */sbin/netreport*. Request notification of changes in the network interface
- */usr/bin/write*. Send messages straight to another user's console
- */usr/bin/wall*. Send messages straight to the consoles of all users on the system

Once the suids above have been disabled, the following suid applications remain:

- *chfn, chsh*. Lets a user change their finger name and shell. Must be suid to make changes to /etc/passwd. You can safely disable this if you don't want user's changing their personal contact information or default shell.

- *newgrp*. Lets users add themselves to a group if they have the appropriate password or are otherwise a member of the group. Must be suid to make changes to /etc/group. You can safely disable this if a user does not need to access group membership by supplying a password.

- *gpasswd, passwd*. Lets users change group and personal passwords. Must be suid to make changes to /etc/gshadow and /etc/shadow. Disable this if you don't want users changing their passwords. However, *you should encourage* users to change their passwords on a regular basis.

- *procmail, sendmail*. Lets Sendmail deliver mail. Must be suid so the application can write to privately owned mailboxes (to deliver mail).

- *su*. Lets people login to other user accounts if they know the password. Must be suid to assume the identity of another UID.

- *crontab*. Scheduled execution of programs.

- *Ping, traceroute*. Various network diagnostic tools. Suid is required because normal users do not have appropriate permissions to access the network in

the manner needed to perform a ping or traceroute. You can safely disable this if you don't want users, other than root, to perform network diagnostics.

## Additional service configurations

If you disabled Sendmail as suggested, but would still like email delivered by (sent *from*) your host, add a cron job (scheduled task). Adding the following line to the root crontab (by executing 'crontab –e' as root) enables outbound mail delivery:

```
0 * * * * /usr/lib/sendmail –q
```

This lets Sendmail attempt queued e-mail delivery every hour. Email is queued only after the initial delivery attempt fails.

If you're running Sendmail to receive incoming email, review and configure it appropriately. Sendmail configuration is beyond the scope of this document. You can review all of the documentation, along with configuration tips and advice at http://www.sendmail.org/. O'Reilly also publishes a comprehensive guide to Sendmail (http://www.oreilly.com/)

**FTP**

If you're running FTP, make sure you disallow FTP access to all system accounts, as well as all other users not allowed to FTP files. Disallow an account access by placing the account name in /etc/ftpusers (you may need to create this file if it doesn't exist). This file contains a list of users, one per line, who are not allowed to use FTP. Minimally you should have root, bin, daemon, adm, lp, sync, shutdown, halt, mail, news, uucp, operator, games, gopher, ftp, nobody, lists and xfs. Make sure this file is secure by running

```
chmod 600 /etc/ftpusers
```

Also, modify the FTP daemon's startup parameters in /etc/inetd.conf to enable more verbose logging. We suggest the following parameters:

*-a* Do not allow users listed in /etc/ftpaccess to use FTP
*-l* Log incoming connections
*-v* Log all user-submitted commands. This helps you monitor what users are doing, at the expense of (many) log entries. Note that this does not log the initial user login (see -L below).
*-L* Log *all* user commands, including the initial log in process. This is useful to detect FTP-based attacks.
*-i* Log all incoming file transfers to /var/log/xferlog
*-o* Log all outgoing file transfers to /var/log/xferlog

In /etc/inetd.conf, your FTP command line should read:

```
ftp   stream  tcp   nowait  root   /usr/sbin/tcpd  in.ftpd -l -a -
v -L -i -o
```

## Miscellaneous

### SSH

SSH (secure shell) is a secure replacement for telnet. SSH encrypts the entire session and prevents passwords and data from being sent on the network in clear text. SSH packages usually include SCP (secure copy), which is a secure file transfer alternative to FTP.

SSH is widely available as a source package. Grab a copy from http://www.freessh.org/.

We recommend using version 1.2.27; the licensing model becomes more strict starting with version 2.0. Your licensing setup may require you to use the RSAREF library. Details about this library are included within the SSH source README.

There's also an open-source version of SSH, called OpenSSH, available from http://www.openssh.com/. OpenSSH uses non-patented cryptographic protocols, resulting in fewer licensing restrictions. However, OpenSSH is not compatible with regular SSH (above), which uses patented cryptographic algorithms.

There are also commercial SSH packages available from http://www.datafellows.com/.

### NMAP

NMAP is a popular system service analysis tool, commonly called a 'port scanner.' Use NMAP to determine what services are being offered publicly to the Internet. You'll find NMAP at http://www.insecure.org/nmap/.

Luckily a RedHat i386 Linux RPM is available to make installation easy. Once installed, invoke NMAP to scan your local system by running:

```
nmap localhost -p 1-65535
nmap localhost -sU -p 1-65535
```

This will look for and list all ports that are listening (labeled as 'Open') on your system. On a minimal, secure system, you would have:

```
Interesting ports on localhost (127.0.0.1):
Port    State       Protocol  Service
22      open        tcp       ssh
80      open        tcp       web
```

You might also have entries for telnet (port 23) and FTP (port 21). The second scan, which has the -sU parameter, tells nmap to check for open UDP ports as well. On a secure system, none should be found, and the result message should look like this:

```
No ports open for host localhost (127.0.0.1)
```

If NMAP lists a port as 'open' for a service you're unfamiliar with, find out which application is using that port and close it if the service is unnecessary.

**Locking rhosts**

Do not use rlogin/rshell/rexec. These are traditionally considered 'unsafe.' However, other applications, such as SSH, still use .rhosts files in combination with an SSH-implementation of rhost authentication. Make sure no root-based rhost authentication is possible; create and lock system rhost files by running the following commands:

```
touch /root/.rhosts /root/.netrc /etc/hosts.equiv
chmod 0 /root/.rhosts /root/.netrc /etc/hosts.equiv
```

This creates empty rhost-related system files and changes their permission disallowing modification.

**Legal banners**

You may want to include a legal warning of usage for the system when users log on. Your warning should be placed in /etc/issue.net and /etc/issue (shown before users log in), as well as /etc/motd (shown after users log in).

**Using development tools**

Strive to keep development tools off production systems. The proper setup includes separate development servers with configurations identical to your production servers. The development servers host all development tools (compilers, etc). In this situation, you build and test applications on the development servers, then transfer the final binaries to the production servers for use.

### TCP_Wrappers

Many entries in /etc/inetd.conf have a '/usr/sbin/tcpd' prefix. The tpcd application is known as 'tcp_wrappers.' It's a security tool that lets you control access to any particular application based on source IP addresses. More information is available in the man pages ('man tpcd'), or in the original application distribution, available at ftp://ftp.porcupine.org/pub/security/index.html.

### IPChains

IPChains let you set up firewall/network filtering rules, to limit access to your system. Using IPChains is beyond the scope of this document. You'll find a complete guide at http://metalab.unc.edu/pub/Linux/docs/HOWTO/IPCHAINS-HOWTO.

### PAM (Pluggable Authentication Modules)

PAM let you control the authentication process and use alternate authentication schemas (NDS, RADIUS, TACACS, etc). More information on using PAM is available at ftp://ftp.us.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam.html.

### OpenWall Project

The OpenWall Project comprises various patches to the Linux kernel for limiting functionality and -- in general – making the system more secure. Find out more about it at http://www.openwall.com/linux/README.

### Bastille Linux Project

A group of developers are working on an application meant to secure default Linux systems. Essentially, you download and run a script and it modifies your system to be 'more secure'. It's still in development, but it's well worth a look at http://www.bastille-linux.org/.