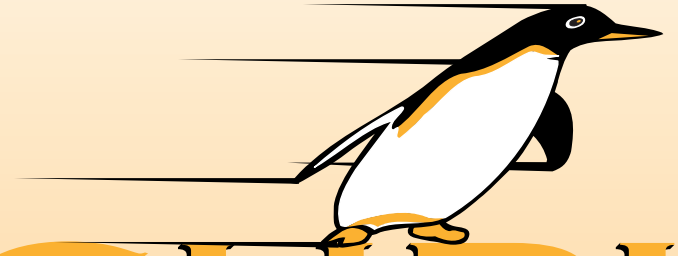
A Survival Guide for Linux Security

A consensus document by security professionals from 46 commercial, educational, and government institutions.



SECURING LINUX

STEP-BY-STEP

VERSION 1.0

THE SANS INSTITUTE

One of the great sources of productivity and effectiveness in the community of computer professionals is the willingness of active practitioners to take time from their busy lives to share some of the lessons they have learned and the techniques they have perfected. Much of the sharing takes place through online news groups, through web postings, and through presentations at technical meetings, and those who are able to take the time to scan the newsgroups, surf the web, and attend the meetings often gain invaluable information from those interactions.

SANS' Step by Step series raises information sharing to a new level in which experts share techniques they have found to be effective. The SANS Institute integrates the techniques into a step-by-step plan and then subjects the plan, in detail, to the close scrutiny of other experts. The process continues until consensus is reached. This is a difficult undertaking. A large number of people spend a great deal of time making sure the information is both useful and correct.

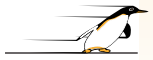
INTRODUCTION

From a small, collaborative effort headed by a University of Helsinki student named Linus Torvalds, Linux has grown into a global phenomenon spurring new industries for distribution, training, and support. It is now the only operating system other than Windows NT that is gaining market share in corporate Information Technology infrastructures. Distributors are actively marketing easier to install, easier to use Linux systems and making real inroads onto the desktops of home, corporate, government, and educational users. By some estimates there are nearly 8 million Linux users worldwide.

Linux is an "Open Source" operating system. The source code for the kernel and system utilities is available for download, inspection, and modification. This is a double-edged sword: system developers and ordinary users alike have access to the source code so bugs are found and fixed more quickly; but system crackers have access to the code as well, and they can use this knowledge to develop exploits more rapidly and reliably. This does not make Linux less secure than its proprietary competition. On the contrary, bugs are discovered faster in an open environment, and patches and updates are issued for Linux system software very quickly. Unfortunately, most users install Linux from CD-ROM media that quite often contains vulnerable programs by the time the ink dries on the label. Another unfortunate aspect of installing commercial Linux distributions is that, for ease of use, these Linux systems are configured with most, if not all, network services running immediately after the computer is booted up, and without any access controls in place. For example, for years all Linux distributions have shipped with TCP wrappers in place, but the `/etc/hosts.allow` and `/etc/hosts.deny` files are empty, meaning that anyone on the Internet can connect to TCP wrapped services.

This guide is intended for the novice home user and the experienced systems administrator alike. It covers the installation and operation of Linux in two basic modes of operation: as a workstation and as a server. It does not cover configuring Linux for some of the other special-purpose functions that it performs so well, such as routers, firewalls, parallel processing, and so forth. The examples and instructions are based on the Red Hat version 6.0 release. Red Hat was chosen because it has the largest share of the Linux market, and version 6.0 was chosen because it includes the latest stable release of the Linux kernel, system libraries, utilities, etc. However, the concepts, advice, and procedures in this guide should translate rather easily to other distributions. You may have to explore your system a little to find configuration files that are in different directories, and to determine which versions of the software packages have been installed, but the exploration itself can be a good instructional tool.

This guide takes you, the reader, through the installation process then splits into separate steps for securing a workstation setup and a server setup. The guide discusses basic packet firewalls in terms of protecting services on a single local computer. Finally, the guide discusses a few useful tools for monitoring and testing the security of your system. We try to follow the principle of "defense in depth." No one step is a silver bullet against system attacks, but taken as a whole, they build multiple layers of defense that make life just that much harder for "script kiddies" and dedicated computer criminals.



INTRODUCTION

We try to explain, as much as possible, the options and ramifications of securing a Linux box. However, this guide is not a text on computer security. Whenever possible, pointers to other documents and resources are included in the text or in Appendix A. We encourage you to study these other resources before setting up your Linux box, and to keep abreast of the latest information from the distributor, the SANS Institute, and other cited security references.

By convention, commands executed by the root user are preceded with the command-line prompt "`[root]#`". In the body of the text, system commands and file names are in `Courier fixed-space font`. Command sequences and file contents are separate from the text, also in Courier fixed-space font. References to manual pages are in the traditional style, page (section), e.g. `inetd.conf(5)`. To read the manual page in Red Hat Linux, and most other distributions, execute the command:

```
[root]# man 5 inetd.conf
```

IMPORTANT:
Updates will be issued whenever a change in these steps is required, and new versions will be published periodically. Please email info@sans.org with the subject <Linux Security> to subscribe to the monthly Network Security Digest containing news of new threats and solutions and announcements of updates. There is no charge.

This edition was guided and edited by:
Lee E. Brotzman, Allied Technology Group, Inc. and David A. Ranch, Trinity Designs

The SANS Institute enthusiastically applauds the work of these professionals and their willingness to share the lessons they have learned and the techniques they use.

Tyler J. Allison, AboveNet Communications, Inc.
Ofir Arkin, LinuxPowered.com
Scott Barker, MostlyLinux, Inc.
Mario Biron, Geonetix Technologies Inc.
Daniel T. Brown, Air Force Research Laboratory, Rome Research Site, Information Assurance Office
David Brumley, Stanford University Network Security Team
Richard Caasi, Science Applications International Corporation
Ian C. Campbell, State University of New York at Albany
John Coleman, Yale University Library Systems
Andrew Cormack, JANET-CERT
Patrick Darden, Athens Regional Medical Center
Clement Dupuis, UniGlobal, Montreal Canada
John F. Feist, Space and Naval Warfare Center
Robin Felix, R. L. Phillips Group
William James Hudson, Robert Mann Packaging, Inc.
Det. Ted Ipsen, Seattle Police Department
Roy Kidder, Corecomm Communications
Alexander Kourakos, Biz Net Technologies
Chet Kress, APAC Customer Services
Loren E. Heal, University of Illinois at Urbana-Champaign
John Lampe, EDMT Technologies
Jonathan Lasser, University of Maryland, Baltimore County (UMBC)
Bill Lavalette, Network Disaster Recovery Systems
Manuel Lopez, Universidad Autonoma de Baja California
Chandrashekhar Marathe, Lucent Technologies India PRC, Bangalore
Rob Marchand, Array Systems Computing
Mike Marney, Robins Air Force Base
John E. Meister, Jr., Intermec Technologies Corporation
Shane B. Milburn, Science Systems and Applications, Inc.
P. Larry Nelson, University of Illinois at Urbana-Champaign
Stephen Northcutt, The SANS Institute
Davi Ottenheimer, M & I Data Services
Jari Pirhonen, AtBusiness Communications Ltd.
Jesse I. Pollard, II, Logicon Information Systems & Services
Patrick O.C. Ramseier, Pilot Network Services, Inc.
Dave Remien, SCIENTECH, Inc.
Andy Routt, Concept Computing
David Saunders, University of Virginia
Kurt Seifried, SecurityPortal. com and Seifried.org
J. J. Shardlow, Tertio, Ltd.
Andres J Silva III, Collective Technologies
Derek Simmel, Software Engineering Institute, Carnegie Mellon University
Len Smith, University of Michigan, College of LSA
Aurobindo Sundaram, Schlumberger IT
Robert Thomas, U.S. Census Bureau
Bob Todd, Advanced Research Corporation

CONTENTS

STEP 1 BEFORE INSTALLATION 1

STEP 1.1: DETERMINE THE SECURITY NEEDS 1
- *Step 1.1.1. Define security policies* 1

STEP 1.2: PHYSICALLY SECURE THE COMPUTER

STEP 1.3: BIOS SECURITY: PASSWORD PROTECTION, LIMITING REBOOTS 2
- *Step 1.3.1. Disable "AUTO" settings* 2
- *Step 1.3.2. Disable booting from removable media* 2
- *Step 1.3.3. Set a BIOS password* 3
- *Step 1.3.4. SCSI BIOS setups* 3
- *Step 1.3.5. Document BIOS settings* 3

STEP 2 INSTALL LINUX 4

STEP 2.1: DISCONNECT THE MACHINE FROM THE NETWORK 4

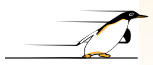
STEP 2.2: SELECT INSTALLATION CLASS: WORKSTATION, SERVER, OR CUSTOM 4

STEP 2.3: DEFINE PARTITIONS 5
- *Step 2.3.1. Define Workstation partitions* 5
- *Step 2.3.2. Define Server partitions* 6
- *Step 2.3.3. Document the partition scheme* 8

STEP 2.4: SELECT PACKAGES TO INSTALL 8
- *Step 2.4.1. Workstation packages* 9
- *Step 2.4.2. Server packages* 9
- *Step 2.4.3. Let the installation proceed* 9

STEP 2.5: CONFIGURE THE SYSTEM SECURITY AND ACCOUNT POLICIES 10
- *Step 2.5.1. Shadow Passwords with MD5 hashing* 10
- *Step 2.5.2. Set passwords for root and all user accounts* 10

STEP 2.6: FINAL LINUX INSTALLATION RECOMMENDATIONS 11
- *Step 2.6.1. Create a boot diskette* 11
- *Step 2.6.2. Tighten up settings in /etc/inittab* 11
- *Step 2.6.3. Password protect LILO boots* 13



CONTENTS

STEP 2.7: SET SYSTEM ACCESS SECURITY POLICIES . . . 12
- *Step 2.7.1. Check that remote root logins are disabled for TELNET* . . . 12
- *Step 2.7.2. Check that remote root logins are disabled for FTP* . . . 13
- *Step 2.7.3. Configure the system accounts that can/cannot log into the system* . . . 13
- *Step 2.7.4. Configure the system groups that can/cannot use specific resources* . . . 13

STEP 2.8: CONFIGURE LOGGING . . . 13
- *Step 2.8.1. Optimize SYSLOG settings* . . . 14
- *Step 2.8.2. Configure real-time logging to VTYs* . . . 14
- *Step 2.8.3. Configure log rotation* . . . 15
- *Step 2.8.4. Configure remote logging* . . . 17
- *Step 2.8.5. Synchronize system clock with log server* . . . 17

STEP 3 SECURING WORKSTATION NETWORK CONFIGURATIONS . . . ***18***

STEP 3.1: DISABLE INTERNET DAEMON SERVICES . . . 18
- *Step 3.1.1. Edit /etc/inetd.conf and comment out all services* . . . 18
- *Step 3.1.2. Turn off inetd if there are no services* . . . 19

STEP 3.2: USE TCP WRAPPERS TO CONTROL ACCESS TO REMAINING INETD SERVICES . . . 19
- *Step 3.2.1. Set the default access rule to deny all* . . . 19
- *Step 3.2.2. Allow access to only specific hosts for specific services* . . . 20
- *Step 3.2.3. Check the syntax of the access lists with tcpdchk* . . . 20
- *Step 3.2.4. Set up banners for TCP wrapped services* . . . 21

STEP 3.3: DISABLE RUN-TIME NETWORK SERVICES . . . 22
- *Step 3.3.1. Determine which network services are running* . . . 23
- *Step 3.3.2. Eliminate unnecessary services* . . . 25
- *Step 3.3.3. Check for any remaining services* . . . 26

STEP 3.4: GET THE LATEST VERSIONS OF SOFTWARE . . . 27
- *Step 3.4.1. Find security-related updates* . . . 27
- *Step 3.4.2. Download updates* . . . 27
- *Step 3.4.3. Install updates* . . . 28
- *Step 3.4.4. Automate the process* . . . 29
  - *Step 3.4.4.1. Use AutoRPM to automate updates* . . . 29
- *Step 3.4.5. Subscribe to security-related mailing lists* . . . 31

CONTENTS

STEP 3.5: CACHING-ONLY DOMAIN NAME SERVICE (DNS) . . . 32
- Step 3.5.1. Disable and remove DNS server software . . . 32
- Step 3.5.2. Set primary and secondary name servers. . . . 32

STEP 3.6: ELECTRONIC MAIL. . . . 33
- Step 3.6.1. Turn off sendmail daemon mode . . . 33
- Step 3.6.2. Define SMTP server for mail clients . . . 33
  - Step 3.6.2.1. Set out-bound SMTP server for sendmail . . . 34
  - Step 3.6.2.1. Set out-bound SMTP server for other mail clients . . . 34

STEP 3.7: NFS CLIENT-SIDE SECURITY . . . 35
- Step 3.7.1. Turn off NFS exports and remove NFS daemons . . . 35
- Step 3.7.2. Configure local NFS mounts . . . 35

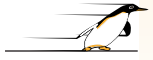
STEP 3.8: LIMIT WORLD WIDE WEB SERVICES TO THE LOCAL HOST. . . . 37
- Step 3.8.1. Turn off HTTP and remove the server software . . . 37
- Step 3.8.2. Limit HTTP access to localhost only . . . 37

STEP 3.9: REMOVE ANONYMOUS FTP SERVICE . . . 38

STEP 4 SECURING SERVER NETWORK CONFIGURATIONS

STEP 4.1: SERVERS: SEE STEPS 3.1, 3.2, 3.3, AND 3.4 FOR DISABLING ALL UNNECESSARY SERVICES, SETTING WRAPPERS, AND UPDATING SOFTWARE . . . 38

STEP 4.2: INSTALL SECURE SHELL FOR REMOTE ACCESS . . . 40
- Step 4.2.1. Download, compile, and install SSH . . . 40
- Step 4.2.2. Start the SSH daemon . . . 41
- Step 4.2.3. Set up /etc/hosts.allow for SSH access. . . . 41
- Step 4.2.4. Generate SSH keys . . . 42
- Step 4.2.5. Use SSH and SCP for remote access . . . 43
- Step 4.2.6. Replace 'r' programs with SSH . . . 45



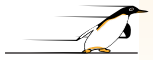
CONTENTS

STEP 4.3: DOMAIN NAME SERVICE AND BIND VERSION 8 45
- *Step 4.3.1. Restrict zone transfers* 45
- *Step 4.3.2. Restrict queries* 46
- *Step 4.3.3. Run named in a chroot jail* 46
  - *Step 4.3.3.1. Create the new user and group* 47
  - *Step 4.3.3.2. Prepare the chroot directory* 47
  - *Step 4.3.3.3. Copy configuration files and programs* 47
  - *Step 4.3.3.4. Copy shared libraries* 47
  - *Step 4.3.3.5. Set syslogd to listen to named logging* 47
  - *Step 4.3.3.6. Edit the named init script* 48
  - *Step 4.3.3.7. Specify a new control channel for ndc* 48

STEP 4.4: ELECTRONIC MAIL 49
- *Step 4.4.1. Turn off SMTP vrfy and expn commands in /etc/sendmail.cf* 49
- *Step 4.4.2. Define hosts allowed to relay mail* 50
  - *Step 4.4.2.1. Check that the access database is active* 50
  - *Step 4.4.2.2. Set access for domains allowed to relay* 51
- *Step 4.4.3. Set domain name masquerading* 51
- *Step 4.4.4. Install an alternative MTA* 52
- *Step 4.4.5. Secure the POP and IMAP daemons* 52
  - *Step 4.4.5.1. Get the latest version of POP and IMAP daemons* 52
  - *Step 4.4.5.2. Control access to POP and IMAP with TCP wrappers* 52
  - *Step 4.4.5.3. Install an alternative POP or IMAP daemon* 53
  - *Step 4.4.5.4. Install an SSL wrapper for secure POP/IMAP connections* 53

STEP 4.5: PRINTING SERVICES 53
- *Step 4.5.1. List allowed remote hosts in /etc/hosts.lpd* 53
- *Step 4.5.2. Replace Berkeley lpr/lpd with LPRng* 53
  - *Step 4.5.2.1. Download and install LPRng* 54
  - *Step 4.5.2.2. Set remote hosts and/or networks that are allowed access* 54

STEP 4.6: NETWORK FILE SYSTEM 54
- *Step 4.6.1. Set access to RPC services in /etc/hosts.allow* 55
- *Step 4.6.2. Limit exports to specific machines with specific permissions* 55



CONTENTS

STEP 4.7: SERVER MESSAGE BLOCK (SMB) SAMBA SERVER . . . 56
- *Step 4.7.1. Get the latest version of Samba* . . . 56
- *Step 4.7.2. Limit access to specific hosts* . . . 56
- *Step 4.7.3. Use encrypted passwords* . . . 56
- *Step 4.7.4. Remove "guest" or anonymous shares* . . . 57
- *Step 4.7.5. Set default file creation masks* . . . 57

STEP 4.8: STEP 4.8. CENTRAL SYSLOG HOST . . . 58
- *Step 4.8.1. Configure syslogd to accept remote log messages* . . . 58
- *Step 4.8.2. Configure log rotation* . . . 58

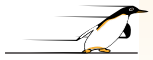
STEP 4.9: FILE TRANSFER PROTOCOL (FTP) . . . 59
- *Step 4.9.1. Limit access with TCP wrappers* . . . 59
- *Step 4.9.2. Limit permitted operations in /etc/ftpaccess* . . . 59
- *Step 4.9.3. Protect incoming directory* . . . 60

STEP 4.10: HYPERTEXT TRANSFER PROTOCOL (HTTP) SERVER . . . 61
- *Step 4.10.1. Set basic access to default deny* . . . 61
- *Step 4.10.2. Selectively open access to specific directories* . . . 61
- *Step 4.10.3. Selectively allow options on specific directories* . . . 62
- *Step 4.10.4. Selectively use .htaccess to override access control* . . . 62
- *Step 4.10.5. Use password protection for sensitive data* . . . 62
- *Step 4.10.6. Use SSL for secure HTTP communications* . . . 63
  - *Step 4.10.6.1. Download OpenSSL and mod_ssl* . . . 63
  - *Step 4.10.6.2. Build OpenSSL* . . . 63
  - *Step 4.10.6.3. Build Apache with mod_ssl module* . . . 64
  - *Step 4.10.6.4. Start Apache with mod_ssl and test* . . . 64
  - *Step 4.10.6.5. Read the mod_ssl documentation* . . . 64

STEP 5 TUNING AND PACKET FIREWALLS . . . ***64***

STEP 5.1: KERNELS: THOUGHTS ABOUT CONFIGURATION, RECOMPILING, AND INSTALLING A NEW KERNEL . . . 65

STEP 5.2: *System optimizations* . . . 66
- *Step 5.2.1. TCP/IP Receive Window size* . . . 66



CONTENTS

STEP 5.3: PACKET FIREWALLS AND LINUX IP MASQUERADING 67
- *Step 5.3.1. Getting more from your external connection with IP Masquerade* 67
- *Step 5.3.2. A strong /etc/rc.d/rc.firewall ruleset* 67
- *Step 5.3.3. Double check, install, and test the firewall* 68
  - *Step 5.3.3.1. Make the ruleset executable* 68
  - *Step 5.3.3.2. Load the ruleset while at the console of the Linux server* 68
  - *Step 5.3.3.3. Test the firewall ruleset* 70
- *Step 5.3.4. Analyze a typical IPCHAINS firewall ruleset hit* 70
- *Step 5.3.5. Running the firewall ruleset upon every reboot* 70

STEP 6 TOOLS 72

STEP 6.1: HOST-BASED MONITORING AND INTRUSION DETECTION 72
- *Step 6.1.1. Swatch, the Simple WATCHer* 72
- *Step 6.1.2. Psionic Logcheck* 73
- *Step 6.1.3. Tripwire* 74
- *Step 6.1.3.1. Tripwire databases* 74
- *Step 6.1.3.2. Running Tripwire* 75
- *Step 6.1.3.3. Use rpm to verify package files* 75
- *Step 6.1.4. Psionic PortSentry* 76

STEP 6.2: HOST-BASED VULNERABILITY ANALYSIS: LOOKING FROM THE INSIDE OUT 77
- *Step 6.2.1. Tiger, the Texas A&M system checker* 77
- *Step 6.2.2. Install and configure Tiger* 77
- *Step 6.2.3. Running Tiger* 78
- *Step 6.2.4. Changing Tiger checks* 79
- *Step 6.2.5 TARA, an updated version of Tiger* 79

STEP 6.3: NETWORK-BASED VULNERABILITY ANALYSIS: LOOKING FROM THE OUTSIDE IN 79
- *Step 6.3.1. SATAN derivatives: SARA and SAINT* 80
- *Step 6.3.2 Nessus* 81
- *Step 6.3.3 Nmap port scanner* 82
- *Step 6.3.4 Commercial products* 83

CONTENTS

APPENDIX A RESOURCES AND REFERENCES 84

APPENDIX B STOCK RED HAT 6.0 **/ETC/INETD.CONF** 87

APPENDIX C RED HAT SYSV INIT SCRIPT FOR THE SSH DAEMON 89

APPENDIX D A STRONG PACKET FIREWALL RULESET 90

APPENDIX E SCRIPT TO MODIFY DEFAULT PERMISSIONS OF SYSTEM BINARIES 110

STEP 1.1 DETERMINE THE SECURITY NEEDS

System security on any operating system needs to be thought out before hand. Security-conscious home computer users and corporations alike need to determine the following before installing:

1. What are you trying to protect? Confidential data (medical records, credit card numbers, etc.), access to essential services (DNS, mail, etc.), computing resources (user shell accounts, software development tools, etc.), and so forth.
2. To what extent should security be implemented before the costs outweigh the value of the original data and/or services being protected?

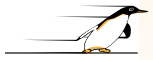
You will need to determine what this machine's main purpose will be. Once determined, try to stick to your original plan. The two primary purposes covered in this guide are:

1. Workstation: X-Windows, user applications like Web browsers, e-mail, word processing, spreadsheets, databases, etc. Workstations don't need to offer services like TELNET, FTP, HTTP, etc. Thus, these services should be shutdown or removed altogether.
2. Server: No X-windows, running network services such as NFS, Samba, HTTP, FTP, SMTP, DNS, shell accounts, firewalls, etc.

Servers should be configured to run as few services as possible. This will offer the administrator better security, fault tolerance, and the ability to scale performance to the services that need it. However, the additional hardware and administrative cost must be considered.

Step 1.1.1 Define security policies

This step is intended more for security professionals and corporate Information Technology managers, but it will be useful to private users concerned about security as well. Corporate and government organizations need to develop, document, and enforce a security and appropriate use policy. The policy should be applied to both internal and external users. Users should learn about the policy through written communication and should sign a document agreeing to the policy. The main reason for a well-documented policy and implementation is that when a security breach or misuse of your systems occur, you have binding legal documents to pursue the people responsible. Without policies and corresponding documentation, enforcement of the policy may be challenged legally.



STEP 1.2 PHYSICALLY SECURE THE COMPUTER

Depending on your security needs, you may have to physically secure the box. Physical security for home users isn't much of an issue, but if you're an ISP that co-locates hardware at another facility, you should think about locking the server(s) up.

- If possible, put the server in a well-ventilated and locked room. Know who has the key to that room.
- Use computer cases that have drive bays and keyboards that can be locked. This ensures that people can't insert floppies and CDs or try to hack away at the console's keyboard.

STEP 1.3 BIOS SECURITY: PASSWORD PROTECTION, LIMITING REBOOTS

Most modern computer BIOSes allow the restriction of bootable devices and password protection for BIOS settings. Different systems use different keystrokes to enter into the BIOS setup but the common ones are the "Del" key for AMI BIOS, "Ctrl + Esc" for Phoenix BIOS, and the "F10" key for Compaq BIOS.

Step 1.3.1. Disable "AUTO" settings

Disable the "AUTO" BIOS settings for options like hard disks, PnP IRQ settings, etc. IDE hard drives configured for "AUTO" configuration can be recognized incorrectly by Linux. System administrators may see file system problems from incorrect BIOS AUTO settings and try to fix the issues with `e2fsck`. This can result in a corrupt file system simply because the IDE "AUTO" setting changed the hard drive translation from the Logical Block Addressing (LBA) mode to the "LARGE" sector layout. Manually configuring the hard drive parameters and other settings ensures that your machine is more reliable and boots faster.

Step 1.3.2. Disable booting from removable media

Enable booting only from the primary hard drive (C:). Do not allow booting from floppy, ZIP, CD-ROM, or any other form of removable media. For medium- and high-security needs, you may want to remove the floppy drive entirely. It should also be noted that some BIOSes allow the floppy drive to be set to READ ONLY, although this will not in and of itself prevent booting the system from a floppy.

Step 1.3.3. Set a BIOS password

Set a BIOS password to ensure that a rogue user cannot change the system's CMOS settings. Use a password with both upper and lower case letters and numbers. See Step 2.5 below for suggestions on selecting strong passwords. There are BIOS password cracking programs available on the Internet so do not use the same password for the BIOS setup to protect LILO (Step 2.6.3) or the root or other user accounts (Step 2.5.2) in the Linux operating system.

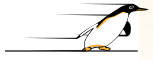
Step 1.3.4. SCSI BIOS setups

Some SCSI controllers have their own BIOS. For example, Adaptec controllers use the Control-A keyboard sequence to enter the SCSI BIOS program. Be sure that you disable the ability to boot off other SCSI media such as CD-ROMs. As of the latest version of Adaptec's SCSI BIOS, there is no way to password protect the SCSI BIOS setup. So if a user can reboot the computer, go into the SCSI BIOS, and enable CD-ROM booting. The only thing keeping them out of your system is a locked CD-ROM cage.

Step 1.3.5. Document BIOS settings

Document the BIOS settings for hardware and system support. In case of a system failure or the administrator is absent, documentation of the hard drive geometry, any non-default system settings, etc., will record the basic initial settings for this system. Place this documentation along with the other hardware manuals in a secure place.

Once your security needs are understood and the security policy documented, it's time to install the Linux operating system. Linux offers a wide array of distributions to choose from and each one has its specific pros and cons. Though it is beyond the scope of this document, we highly recommend you give your choice of Linux distribution some thought as it will impact future administration and ease of use. For this guide, we chose Red Hat 6.0 primarily because of Red Hat's market share and our goal to cover as many users as possible. If you are going to use another distribution or an earlier version of Red Hat, you should be aware of the differences in system administration and security-related configuration. For example, older versions of Red Hat do not use shadow passwords by default, system files may be in different directories, Slackware does not use the SYSV init style startup, etc. Whichever distribution you choose, upgrade to the latest version to get all the benefits of better security, functionality, and performance.



STEP 2
Install Linux

STEP 2.1 DISCONNECT THE MACHINE FROM THE NETWORK

Most Linux distributions bring up their network interfaces during installation even though there is minimal system security in place. In light of this, the best security practice is to disconnect the network card before starting the installation. If you are installing from a network server, make sure that the network you are using is secure while the installation is running. If you are installing from a public FTP site on the Internet, be careful, since your system might be compromised before you ever realize it.

STEP 2.2 SELECT INSTALLATION CLASS: WORKSTATION, SERVER, OR CUSTOM

The Red Hat installation program allows for selecting from one of these three choices. The “Workstation” and “Server” selections do automatic partitioning of the system’s hard disks based upon percentages of the overall hard disk size. In addition, these two settings install many software packages that will probably never be used by you or your users. Use the “Custom” installation option for maximum control over the installation process, regardless of whether you are setting up a workstation or server.

Partitioning is a fairly religious debate among UNIX administrators. Why? The theory is that the more partitions a system has, the more reliable it will be. For example, if a partition becomes corrupt or a denial of service attack fills a partition with log messages, the problem is isolated to only that one partition.

It is generally agreed that Linux Workstations only need three partitions: the root partition, "/"; the user partition, "/home"; and a swap partition for virtual memory. Partition schemes for servers depend on the type of service. The general rule for determining the size of a swap partition is to allocate two times the amount of physical memory. For example, a computer with 64 MB of RAM would allocate at least 128 MB of swap space. Systems with limited RAM should allocate relatively more swap space, systems with a large amount of RAM should allocate relatively less. Use common sense. A system with 1GB of RAM doesn't need 2GB of swap. A system with 32 MB of RAM or less should be upgraded with more memory first. At today's prices for physical memory, adding RAM is the most cost-effective upgrade for your system. Administrators of Linux machines running the older 2.0 version of the kernel should note that this version allows for only 128 MB swap partitions. For more swap space, multiple partitions are required. The latest releases of most Linux distributions, including Red Hat 6.0, use the newer 2.2 version of the kernel, which does not have this restriction.

Step 2.3.1 Define Workstation partitions

Typical workstation installations with the X-Window system, window managers, development tools (if necessary), home office applications (if necessary), Web browsers, and core system utilities can consume up to 700-800 MB of disk space. A minimum root partition of 1 GB will leave room for system logs, spooled email messages, and other housekeeping data. Consider even more space for the root partition if the space is available and the requirement for space on the user's (/home) partition is not that great. For workstations that will make extensive use of logging, consider a separate, relatively large partition for /var.

As an example: a mid-range PC with 64 MB of RAM and a 4 GB hard drive could be partitioned like so:

```
/        1800 MB
swap      200 MB
/home    2000 MB
```

Note that if your workstation is going to "dual-boot" Linux and another operating system, the Linux root partition needs to be within the first 1024 cylinders of the boot disk. This is because LILO, the Linux Loader, is limited to using the BIOS to access the disk and the BIOS can not read past 1024 cylinders. See the Installation HOWTO cited in Appendix A for more information. Of course, your mileage may vary but give this topic some serious thought.

Step 2.3.2 Define Server partitions

Disk partitioning for servers is a very different animal. In general, **all** servers benefit from a separate, relatively large partition for `/var`, for logging, configuration, and temporary space. Additional factors that need to be considered depend on the purpose for the server:

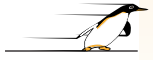
NNTP: News servers require very large amounts of spool space, so a majority of disk space should be reserved for NNTP use only. Even better, put the news spool on its own drive(s) for increased disk performance. Red Hat puts NNTP spool files in `/var/spool/news`.

HTTP: Web servers require disk space for the site's main Web pages as well as users' individual Web pages (if allowed). Web servers should leave plenty of space for HTTP log files which can consume quite a bit of space depending on how logging is configured, how long the logs are retained, etc. Red Hat puts the main Web pages in `/home/httpd` and typically, individual user Web pages go in the user's own home directory under `/home` in the `public_html` directory. Red Hat puts HTTP log files in `/var/log/httpd`.

FTP: FTP servers can require anything from marginal disk space to very large disk space depending on how they are used. Note that if you are going to allow incoming FTP services, there should be enough disk space, or even a dedicated disk, to allow for large sets of files to be uploaded. The main FTP directory for Red Hat is located in `/home/ftp` though users can "cd" into their home directory to get or put files. Red Hat puts FTP log files in `/var/log/xferlog`.

NFS/Samba: File servers typically require a lot of disk space unless the data is directly mounted on CDROM changers, other external jukeboxes or RAID disks. These systems should be able to scale with the needs of the users. Samba typically uses the user's home directory in `/home/*` to put and get files but NFS can be configured many different ways.

E-mail: Mail servers can require large amounts of spool space for storing incoming and outgoing messages, file attachments, etc. Red Hat puts incoming mail in `/var/spool/mail` in files uniquely named for each mail user account. Outgoing mail is stored in `/var/spool/mqueue`.



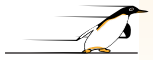
Syslog: A centralized logging host collects and stores syslog messages from other computers in the organization. As you can imagine, logging hosts need large amounts of disk space reserved for the `/var/log` directory. Good practice is to place `/var/log` on a large, fast SCSI device. Logging hosts should not run any other services at all, and should be very secure, so the amount of disk space required for the system software is smaller than for other servers.

Shell: Unix servers that provide shell accounts should have separate partitions for `/tmp` and `/var/log`. This will reduce the impact of classic denial of service attacks that fill the `/tmp` directory with files or create a process that quickly fills the system logs until the root partition is full. Once the root partition is full or the system cannot write any more log files, the system will usually crash.

We recommend that servers have separate partitions for `/var` (or even separate partitions for `/var/log` and `/var/spool`), `/tmp`, the root partition "/", and the user partition `/home`, at a minimum. Some administrators might also consider creating partitions for `/usr` which contains the majority of the Linux Operating System, and `/usr/local` for optional user-installed applications and tools. Some distributions store large system packages like KDE, Netscape and DBMS systems in the `/opt` directory tree, so you might consider a separate partition for that, too. The relative sizes of these partitions are solely at the discretion of the administrator depending on the purpose of the server and your own personal experience.

As an example: a high-end PC with 256MB of RAM and (2) 9GB hard drives Primarily used for NNTP news services could be partitioned like so:

```
First hard disk    /                500MB
First hard disk    /usr            1000MB
First hard disk    /usr/local      1500MB
First hard disk    /tmp            2000MB
First hard disk    /var/spool/news 4000MB

Second hard disk   swap             500MB
Second hard disk   /home           5500MB
Second hard disk   /var            3000MB
```



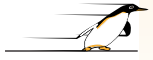
Step 2.3.3. Document the partition scheme

Once the system has been partitioned for your specific environment, document the partition layout. Then, if a partition table is somehow corrupted or deleted, recovering the table will be easier. To document the various tables, run fdisk on every one of your physical disks (hda, sda, etc) and enter "p" to print the table. Copy the results onto paper and store in a safe place for later reference. For example:

```
[root]# fdisk /dev/hda

Using /dev/hda as default device!

Command (m for help): p

disk /dev/hda: 255 heads, 63 sectors, 2193 cylinders
Units = cylinders of 16065 * 512 bytes

Device     Boot   Start     End      Blocks    Id    System
/dev/hda1    *        1    1913    15366141    83    Linux
/dev/hda2          1914    1946      265072+   82    Linux swap
/dev/hda3          1947    2193     1984027+   83    Linux

Command (m for help): q
```

STEP 2.4 SELECT PACKAGES TO INSTALL

Like setting disk partitions, the packages chosen for installation depend on the use of the system. The best way to get through this section is to know your users and how the system will be used. Determine what will and won't be done on the system and choose the software packages accordingly. Carefully go through each section of the installer and hit "F1" whenever you don't know what a given package is or if you aren't sure if you need it or not. When in doubt, leave it out. It's easy enough to add the package later as the need arises. If you don't select a package that Red Hat needs for some other tool, it will warn you about any software dependencies before the installation process actually begins.



Step 2.4.1. Workstation packages

Workstation users usually can fit into one of three categories:

1. Business. The system is used for business purposes like word processing, spreadsheets, E-mail, and WWW browsing. For these systems, there is no need to install services like NFS or HTTP servers, or compilers like GCC and EGCS.
2. Software development. For systems like this, package selection depends on the type of development. A Web developer may need a Web server, PHP, and Perl. Software developers may need GCC or EGCS, G++, Python, Perl, etc., and documentation tools like TeX and Ghostscript.
3. General Purpose. This category probably fits the needs of most home users. Business tools, development tools, multimedia players, games, graphical utilities, are all loaded on the system. The general-purpose user doesn't want to be limited to the fact that they don't have a Web server or GCC installed on their machine. Though this kind of installation is the most convenient for the end user, it makes the maintenance and security of the Linux machine more difficult.

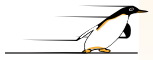
Step 2.4.2. Server packages

Servers are configured for utility, performance, and security. If your budget permits, we recommend you put information services such as WWW and FTP, file and print services such as NFS and Samba, DNS services, email services, and user shell account services on different servers. For example, an information server would only install the base system, FTP and WWW services, Perl, PHP, and nothing else. If you need to have both information services and file services running on the same machine, so be it, but try to minimize the installation of any other unnecessary software.

Step 2.4.3. Let the installation proceed

Once you have selected all the packages you want installed on this system, the Red Hat installer will figure out the dependencies between the packages you selected. It will notify you of any additional packages that you need to install to properly support the selected packages. At this point, you can allow the installer to include the dependant packages or remove the original selection from the installation list entirely.

Finally the installer will load the selected packages and prompt you for any additional configuration settings, such as network cards, mice, video cards, etc. Covering these steps is beyond the scope of this guide but it is provided in the documentation supplied with the shrink-wrap versions of the distributions or on your Linux distribution's web site.



Step 2.5.1. Shadow Passwords with MD5 hashing

One of the final installation options that Red Hat prompts the user with is whether to use "shadow" passwords with or without MD5 cryptographic hashes. Traditionally, users' UNIX passwords were encrypted with the "crypt" algorithm and stored in the world-readable file, `/etc/passwd`. This makes it easy to perform dictionary attacks using tools such as Crack and its cousins. Shadow passwords are stored in `/etc/shadow`, readable only by `root`, which makes it much harder to obtain the encrypted passwords for a dictionary attack. Use the MD5 cryptographic hash to add another layer of security to the shadow password system. It is very important to keep `/etc/shadow` secure and away from prying eyes.

Step 2.5.2. Set passwords for root and all user accounts

One of the last steps before the system reboots is setting the root password. Linux passwords are case sensitive and the installer will recommend that a root password be a combination of UPPER and lower case letters, digits, and special characters including:

```
[ '~!@#$%^&*()-_=+{[]}\|'";:,<.>/?  ]
```

The password should **not** be based on any personal information such as name, company, Social Security number, birthday, license plate number, etc. A determined attacker can find out a surprising amount of personal information about users, especially in the case of an insider attack.

Never use a common word that would be found in any dictionary in any language, like the English word "ridge". Using digits to substitute for letters, for example "r1dg3", does not help, since common password cracking tools take this trick into account.

So what is a good password? Be unique and do combinations; think of a phrase and make an acronym (don't use an acronym from your everyday work, though). For example, a good password can be constructed from the names of the computer science algorithms for B+ and Patricia trees, "`B+paTs.!!`". Be creative but use something you can remember!

It is an excellent practice to teach your fellow system administrators and users alike how to choose strong passwords. With strong passwords that are changed on a frequent basis, systems are much more difficult to break into.

Step 2.6.1. Create a boot diskette

One step that many Linux users skip is the creation of an emergency boot diskette. The Red Hat installer creates a bootable diskette with a Linux kernel with all of the specific hardware and configuration support for your computer. Though creating a boot disk isn't a required step, eventually, a system emergency will come up you'll be happy you created it. Make sure the boot diskette is properly labeled and stored in a secure place like a tape cabinet in a computer room or locked desk drawer.

Although we strongly suggest you create the emergency diskette **now**, if you decide to do it later, in Red Hat Linux the script `/sbin/mkbootdisk` will create the disk for you. See the man page `mkbootdisk(8)` for more information.

Step 2.6.2. Tighten up settings in `/etc/inittab`

The file `/etc/inittab` defines the boot behavior of the SYSV `init` process, process number 1 (see Step 3.3 for additional information about SYSV init startup scripts). After the first reboot, edit `/etc/inittab` to disable rebooting from the console with the "Control + Alt + Del" key sequence, and to require entering the root password to enter single user mode.

To disable the "Control + Alt + Del" key sequence, change the line in `/etc/inittab` file that reads:

```
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

to read:

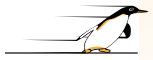
```
#ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

To require entering the root password when you enter single user mode, add the following line to `/etc/inittab` after the entry for `si::sysint...`:

```
~~:S:wait:/sbin/sulogin
```

To make these changes take effect immediately, type in the following command:

```
[root]# init q
```



Step 2.6.3. Password protect LILO boots

The Linux Loader (LILO) is the primary mechanism for booting Linux. If the physical security of the Linux machine can not be assured, password protect the LILO prompt in addition to requiring the root password to enter single user mode. Note that password protection of the LILO prompt may interfere with the automatic restart of a system after a power failure or system crash. Administrators of mission-critical servers in controlled computer room environments may not want this feature.

To password protect the LILO prompt, edit `/etc/lilo.conf` and add the following after the `prompt` line:

```
password = your-password
restricted
```

Replace "`your-password`" a strong password as shown in Step 2.5.2. Save `/etc/lilo.conf` and change the permissions of that file since the LILO password is saved in clear text:

```
[root]# chmod 600 /etc/lilo.conf
```

Make the changes take effect:

```
[root]# /sbin/lilo
```

With these changes in place, the system will boot just has as it has before, but if you give LILO any command line options at the LILO prompt, such as `single` to boot in single-user mode, it will ask for a password before proceeding.

STEP 2.7 SET SYSTEM ACCESS SECURITY POLICIES

Most Linux distributions allow for configuring which users are allowed to login on the system, at which times, on which date, and through which services (TELNET, FTP, etc). Most Linux distributions are configured correctly, if somewhat loosely, out of the box, but it is important to confirm all of these settings.

Step 2.7.1. Check that remote root logins are disabled for TELNET

The `/etc/securetty` file contains a list of all TTY interfaces that allow root logins. In Red Hat Linux, and most distributions, this file by default contains **only** the console TTYs (`tty1, tty2, ... tty8`, inclusive). If remote users require root privileges on this system, they can simply login to their own account and use the "`su -`" command to become root.

Step 2.7.2. Check that remote root logins are disabled for FTP

The `/etc/ftpusers` file contains a list of all user accounts that are not allowed to login via FTP. Make sure that the root account and all additional system daemon accounts are listed in this file. You can also enter account names for other user accounts that you do not want to access the machine by FTP.

Step 2.7.3. Configure the system accounts that can/cannot log into the system

The file `/etc/security/access.conf` contains login parameters for all accounts on the system. You can configure which users can login through which service in a manner similar to TCP wrappers, as described in Section 3.2. To disable all console logins except for the root account and the administrator, put the following in `/etc/security/access.conf`:

```
-:ALL EXCEPT root admin :console
```

Step 2.7.4. Configure the system groups that can/cannot use specific resources

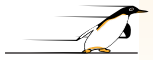
The file `/etc/security/group.conf` defines which users can access files with specific group permissions.

The file `/etc/security/limits.conf` defines system resource limits (CPU time, disk space, etc) for specific users or groups.

The file `/etc/security/times.conf` defines the times of day that specific users are allowed to access the system. Ordinary users can be restricted to logging in during normal business hours through specific network services.

STEP 2.8 CONFIGURE LOGGING

Red Hat and most other distributions use the `sysklogd` package for generating system logs. This package contains two log daemons, `syslogd` and `klogd`. The kernel log daemon, `klogd`, takes care of logging kernel messages and is configured through `/etc/syslog.conf`, just like `syslogd`, and the two work in conjunction with each other seamlessly. Therefore, when we refer to `syslogd`, we are also talking about `klogd`. See the man page `sysklogd(8)` for more information



The syslog daemon parses and separates log messages based on a system of "Facility" and "Level." The stock syslog daemon has some security problems, most notably there is no built-in way to tell if the logs have been tampered with. There are several alternative log daemons that add features such as digital signatures for log files to detect tampering, and encrypted network communications for secure remote logging. See Appendix A for a list of some of these alternative log daemons.

Step 2.8.1. Optimize SYSLOG settings

The default logging for Red Hat Linux is good but it can be made better. To improve it, edit the syslog configuration file, `/etc/syslog.conf`, and add the following lines:

```
*.warn;*.err     /var/log/syslog
kern.*           /var/log/kernel
```

Note: the separators between the syslog facility directives in the first column and the syslog files in the second column ***must*** be TABs. If they are SPACES, the syslog daemon will not load properly.

Step 2.8.2. Configure real-time logging to VTYs

The real-time display of system logs on VTY 7 and 8 (the Alt-F7 and Alt-F8 screens) is very useful. To do this, append the following to `/etc/syslog.conf`:

```
*.info;mail.none;authpriv.none     /dev/tty7
authpriv.*              /dev/tty7
*.warn;*.err            /dev/tty7
kern.*                  /dev/tty7
mail.*                  /dev/tty8
```

Once `/etc/syslog.conf` has been updated, you will need to create the new syslog files and fix their permissions:

```
[root]# touch /var/log/syslog /var/log/kernel
[root]# chmod 700 /var/log/syslog /var/log/kernel
```

To get syslog to immediately use the new configuration file, issue the following command:

```
[root]# killall -HUP syslogd
```

Step 2.8.3. Configure log rotation

Red Hat Linux, and some other distributions, uses the `logrotate` tool to keep log files to manageable sizes and retain them for a specified period of time. See the manual page `logrotate(8)` for more information about the command and the configuration file.

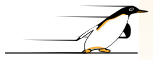
To tune the default Red Hat log rotation configuration and rotate the newly created `syslog` and `kernel` logs in Step 2.8.1, edit `/etc/logrotate.d/syslog` and add the following:

```
/var/log/kernel {
    compress
    postrotate
        /usr/bin/killall -9 klogd
        /usr/sbin/klogd &
    endscript
}

/var/log/syslog {
    compress
    postrotate
        /usr/bin/killall -HUP syslogd
    endscript
}
```

Next, edit `/etc/logrotate.conf` to modify the log rotation behavior to meet your specific needs. In the following example, `logrotate` will rotate logs weekly, keep the last four weeks of logs before it deletes them, compress the rotated logs, and **not** rotate the utmp and wtmp files:

```
# see "man logrotate" for details
# rotate log files weekly
weekly
# keep 4 weeks worth of backlogs
rotate 4

# send errors to root
errors root
# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

## no packages own lastlog or wtmp — we'll rotate them here
#/var/log/wtmp {
#    compress
#    monthly
#    rotate 1
#}

#/var/log/lastlog {
#    compress
#    monthly
#    rotate 1
#}
```

Once these changes are done, they will be automatically executed from the crontab file `/etc/cron.daily/logrotate` once a day (the default setting of Red Hat 6.0).



Step 2.8.4. Configure remote logging

In an organization of any significant size, using a centralized logging host has a lot of benefits. From a security point of view, remote logging is important because one of the first things that an attacker will do after gaining root access to a machine is erase or modify the system logs to cover his tracks. If copies of those log messages are stored on another, highly secured system, administrators may still be able to reconstruct what happened. If the logs are lost, important forensic evidence from the break-in is lost with them.

To send syslog messages to a central logging host, edit `/etc/syslog.conf` and add lines for the facilities and logging levels dictated by the security policy of your organization. For example, to send all warning and error messages, and all authentication facility messages to the central (logging host), called loghost add the following:

```
*.warn;*.err            @loghost
authpriv.*;auth.*       @loghost
```

Step 2.8.5. Synchronize system clock with log server

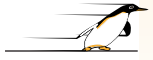
Hardware clocks in PC-class computers are notoriously inaccurate. When many workstations are feeding log messages to a central server, it is important that their clocks are synchronized, so that the time stamps on the log messages are accurate. In the event of a network-wide scan or attack, synchronized log file time stamps will make it easier to build a timeline of the attack. The Network Time Protocol was developed to accurately synchronize the system clocks of a network of computers. An easy way to set the system clock once each hour is to set up a cron job to run `ntpdate` once an hour. Put the following commands into the file `/etc/cron.hourly/set-ntp`:

```
/usr/sbin/ntpdate -bu -t 3 loghost.example.org
/sbin/hwclock --systohc
```

The “b” option sets the time now, rather than staggering it, “u” says to use an unprivileged port, and “t 3” set sets the timeout to 3 seconds, rather than the default 1 second. The second line sets the hardware clock CMOS value to the current system time. Once the file has been saved, set the permissions:

```
[root]# chmod 700 /etc/cron.hourly/set-ntp
```

Of course, this assumes that loghost.example.org is running the `xntpd` NTP daemon. The xntp3 RPM package includes extensive documentation in HTML format in the `/usr/doc/xntp3*` directory. Also see http://www.eecis.udel.edu/~ntp/ for the latest information about NTP and a list of public primary and secondary time servers.



Securing workstations is primarily a process of eliminating network services, and severely limiting access to the remainder. A typical workstation operates on the client end of client-server network communications. There is almost never a need to provide a service. On the remote chance that a service is needed on a workstation, it should be offered only to a small, select group of other workstations or servers on the local area network. In Step 3, we concentrate on removing services, then offer some advice on how to increase the security of selected services if they are absolutely necessary for the operation of the workstation.

STEP 3.1 DISABLE INTERNET DAEMON SERVICES

The Internet daemon, `inetd`, controls access to network services that only start up when needed, such as telnet and ftp. A typical workstation has no need to provide the services in the Internet daemon configuration file, `/etc/inetd.conf`. If you need to access the workstation remotely over the network, consider installing Secure Shell (SSH; see Step 4.2 below). See the manual page `inetd.conf(8)` for a complete description of the format of entries in `/etc/inetd.conf`.

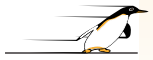
Step 3.1.1. Edit `/etc/inetd.conf` and comment out all services

See Appendix B for a copy of the default `/etc/inetd.conf` installed with Red Hat Version 6.0. The services that are open are those that do not have a hash symbol "#" in column 1. Edit these lines and comment them out by putting the comment character, "#", in the first column. Save the edited file and make the `inetd` process reload the configuration file by sending it the hangup signal:

```
[root]# killall -HUP inetd
```

If you don't comment out any other services, at least make sure that the shell and login services are disabled. The Berkeley "r" programs `rsh` and `rlogin` have a long history of abuse and security problems. Even with the services disabled, you should check to make sure that no user account has a `.rhosts` file in its home directory and that there is no `/etc/hosts.equiv` file. These are the files that the "r" programs use to assign "trust" to users on other computers. The following short shell script will check every home directory and tell you which ones have `.rhosts` files in them.

```
#!/bin/sh
for i in `cut -d: -f6 /etc/passwd`; do
     if [ -e $i/.rhosts ]; then
          echo "SECURITY CHECK found .rhosts in $i"
     fi
done
```



Step 3.1.2. Turn off `inetd` if there are no services

If none of the services in `/etc/inetd.conf` are needed at all, the `inetd` process itself can be turned off:

```
[root]# /etc/rc.d/init.d/inet stop
[root]# /sbin/chkconfig inet off
```

Administrators for distributions that do not have `chkconfig` can simply remove the init scripts directly:

```
[root]# /bin/rm /etc/rc.d/rc[345].d/*inet
```

If, for some reason, one of the services controlled by `inetd` is needed later, it can be turned on temporarily by running the SYSV init script:

```
[root]# /etc/rc.d/init.d/inet start
```

STEP 3.2 USE TCP WRAPPERS TO CONTROL ACCESS TO REMAINING INETD SERVICES

If you absolutely have to have one or more of the services in `/etc/inetd.conf`, use TCP wrappers to restrict access to a limited number of hosts. The TCP wrapper daemon, `/usr/sbin/tcpd`, is a small program that is invoked by `inetd` instead of the normal daemon program. It checks the source address of the request against its access control lists in `/etc/hosts.allow` and `/etc/hosts.deny`. If the lists say that the requester is allowed to connect, the wrapper daemon passes the connection to the appropriate service and everything proceeds as usual. If the lists say to deny access to the service, the connection is dropped. Either way, the wrapper daemon puts an entry in the system logs to document the event.

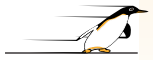
Wrappers are installed by default with Red Hat and all other Linux distributions. The TCP wrapper daemon is invoked by default in `/etc/inetd.conf` for every external service, with the exception of the authentication daemon, `identd`, and the `linuxconf` administration tool. See the manual page for `hosts_access(5)` for details on the syntax of the `/etc/hosts.allow` and `/etc/hosts.deny` files.

Step 3.2.1. Set the default access rule to deny all

Edit `/etc/hosts.deny` so that the **only** uncommented lines read:

```
ALL: ALL
ypserv: ALL
```

The second line is necessary only if you have installed the Network Information System (NIS) ypserv package.



Step 3.2.2. Allow access to only specific hosts for specific services

For example, to allow ftp access from `host.example.org`, edit `/etc/hosts.allow` and add the line:

```
in.ftpd: host.example.org
```

After entering the access control statement in `/etc/hosts.allow`, edit `/etc/inetd.conf` and remove the comment character from the line for the ftp service:

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

and restart `inetd`:

```
[root]# killall -HUP inetd
```

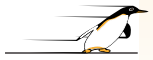
Whenever possible, avoid allowing access to entire domains or subnets, e.g. "`.example.org`" or "`192.168.1.`". Use only the specific hostnames or IP addresses of the machines that are allowed access.

Step 3.2.3. Check the syntax of the access lists with tcpdchk

The TCP wrapper package includes a simple program for checking the syntax of the `/etc/inetd.conf`, `/etc/hosts.allow` and `/etc/hosts.deny` files to make sure that there are no errors and that the files are consistent. Assuming `/etc/hosts.allow` and `/etc/hosts.deny` appear as they do above, here is a sample run of `tcpdchk`:

```
[root]# tcpdchk -v
Using network configuration file: /etc/inetd.conf

>>> Rule   /etc/hosts.allow line 6:
daemons:   in.ftpd
clients:   host.example.org
warning:   /etc/hosts.allow, line 6: host.example.org: host not found
access:    granted

>>> Rule   /etc/hosts.deny line 9:
daemons:   all
clients:   all
access:    denied
```

Note the warning about `host.example.org` saying that it was not found in a name lookup. This tells you that you may have mistyped or otherwise erroneously entered the host name (in this case the hostname is fictitious). Note also that `tcpdchk` will issue warnings about other services that are **not** listed in `/etc/inetd.conf`, but are compiled with TCP wrappers support, such as `portmap` for RPC services (see Step 4.6.1) and `sshd` for Secure Shell (see Step 4.2.3). These warnings can be ignored.



Step 3.2.4. Set up banners for TCP wrapped services

An important option for use with TCP wrappers is that of "banners." Most government agencies, and many commercial organizations, require banners on system access points, such as TELNET and FTP ports. Login banners warning that a system is off limits to unauthorized personnel are often required in order to successfully prosecute a person that breaks into the computer. Some TCP wrapped services can be set up to automatically display a standard login banner. A `Makefile` has been supplied with the TCP wrappers documentation that makes maintenance of banners easier. To create an initial set of banners, execute the following:

```
[root]# mkdir /etc/banners
[root]# cp /usr/doc/tcp_wrappers-7.6/Banners.Makefile /etc/banners/Makefile
[root]# cd /etc/banners
[root]# echo "This is only the prototype banner. Replace with the real banner." >prototype
[root]# make
```

By default, the `Makefile` creates banner files for the `in.telnetd`, `in.ftpd`, and `in.rlogind` services. Read the comments in the `Makefile` for information about banners for other services. The actual wording for the banner is usually a matter of local security policy. The Computer Incident Advisory Capability (CIAC) group has issued an information bulletin about banners that contains sample wording. See Appendix A for the URL.

To activate the banners, edit `/etc/hosts.allow`. For all lines that allow access for the banner services, add the "`banners /etc/banners`" option, followed by the "`allow`" option. Add another option line at the very bottom of `/etc/hosts.allow` setting up banners for all denied services. For example, if `telnet` sessions are allowed from `work.example.org`:

```
in.telnetd: work.example.org : banners /etc/banners : allow
all : all : banners /etc/banners : deny
```

There are a rich set of options for TCP wrappers, including adjusting syslog facilities and levels, adjusting the run-time characteristics of the daemons, "booby traps," and more. See the manual page `hosts_options(5)` and Appendix A for more information.

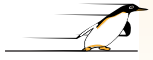
STEP 3.3 DISABLE RUN-TIME NETWORK SERVICES

The Internet daemon handles transient, "one-time" network connections, but there are other network daemons that start at boot-time and normally run until system shutdown, such as `httpd` for the Web, `named` for DNS services, and `smbd` for Samba SMB networking services. They are started at boot-time by the `init` process — process number 1. Red Hat Linux, and most other distributions except Slackware, have a SYSV-style `init`, which uses the concept of runlevels and run-time control scripts to manage the startup and shutdown of services and processes. See Step 2.6.2 for a discussion of the configuration file for `init`, `/etc/inittab`.

See the manual page `init(8)` for more information about `init` and a description of the six normal runlevels. There are two runlevels for normal operation: runlevel 3 has a character-mode console for login; runlevel 5 starts the X Window server and uses the X Display Manager (xdm) login. The scripts that control the startup and shutdown of processes for runlevel 3 are found in the directory `/etc/rc.d/rc3.d`, and runlevel 5 in `/etc/rc.d/rc5.d`. To be precise, the scripts in these directories are symbolic links to general-purpose scripts used for every runlevel in `/etc/rc.d/init.d`.

The names of the scripts are prepended with numbers indicating the order in which they should be executed, for example `S10network` is executed before `S55named`, which is executed before `S85httpd`, and so on. The "S" tells `init` that the script is for startup, a "K" tells `init` that the script is for shutdown (or "Kill"). "S" scripts are passed a single argument, `start`, the "K" scripts are passed the argument `stop`. At any time, root can start or stop services by executing the scripts in `/etc/rc.d/init.d`, for example, to shutdown the Web server:

```
[root]# /etc/rc.d/init.d/httpd stop
```



Step 3.3.1. Determine which network services are running

Use `/bin/netstat` to list the ports that are "listening" for connections:

```
[root]# netstat -at
Active Internet connections (servers and established)
Proto  Recv-Q  Send-Q Local Address            Foreign Address       State
tcp         0       0 *:netbios-ssn           *:*                   LISTEN
tcp         0       0 *:www                   *:*                   LISTEN
tcp         0       0 *:smtp                  *:*                   LISTEN
tcp         0       0 *:1040                  *:*                   LISTEN
tcp         0       0 *:702                   *:*                   LISTEN
tcp         0       0 *:697                   *:*                   LISTEN
tcp         0       0 *:692                   *:*                   LISTEN
tcp         0       0 *:673                   *:*                   LISTEN
tcp         0       0 *:printer               *:*                   LISTEN
tcp         0       0 192.168.1.3:domain      *:*                   LISTEN
tcp         0       0 localhost:domain        *:*                   LISTEN
tcp         0       0 *:sunrpc                *:*                   LISTEN
```

Under "Local Address" the port numbers are reported by name if the service is found in the file `/etc/services`, otherwise only the port number is given. As we can see, the default installation still has a lot of services running even after commenting out all the services in `/etc/inetd.conf`.

Unfortunately, `netstat` does not say which process is listening on which port. There is another tool called `lsof`, for "List Open Files" which lists all the open files, including network sockets, for any process. Depending on the installation process followed in Step 2 above, `lsof` may not be installed. For Red Hat Linux the package is on the installation CD-ROM, or the source code can found on the primary archives listed in Appendix A section. `Lsof` has a lot of options and a lot of flexibility. See the manual page `lsof(8)` for more information. To list all open network sockets:

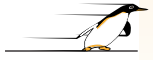
```
[root]# lsof -i +M
COMMAND    PID  USER     FD   TYPE    DEVICE      NODENAME
portmap    3264 root     3u   inet    4546        UDP *:sunrpc[portmapper]
portmap    3264 root     4u   inet    4547        TCP *:sunrpc[portmapper] (LISTEN)
named      3405 root     4u   inet    4724        UDP *:1031
named      3405 root     20u  inet    4720        UDP localhost:domain
named      3405 root     21u  inet    4721        TCP localhost:domain (LISTEN)
named      3405 root     22u  inet    4722        UDP 192.168.1.3:domain
named      3405 root     23u  inet    4723        TCP 192.168.1.3:domain (LISTEN)
lpd        3432 root     5u   inet    4757        TCP *:printer (LISTEN)
rpc.statd  3462 root     3u   inet    4790        UDP *:671[status]
rpc.statd  3462 root     4u   inet    4793        TCP *:673[status] (LISTEN)
rpc.rquot  3472 root     3u   inet    4813        UDP *:681[rquotad]
rpc.mount  3482 root     3u   inet    4833        UDP *:690[mountd]
rpc.mount  3482 root     4u   inet    4836        TCP *:692[mountd] (LISTEN)
rpc.mount  3482 root     5u   inet    4841        UDP *:695[mountd]
rpc.mount  3482 root     6u   inet    4844        TCP *:697[mountd] (LISTEN)
rpc.mount  3482 root     8u   inet    4849        UDP *:700[mountd]
rpc.mount  3482 root     10u  inet    4852        TCP *:702[mountd] (LISTEN)
sendmail   3522 root     4u   inet    4924        TCP *:smtp (LISTEN)
httpd      3549 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3558 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3559 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3560 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3561 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3564 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3565 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3566 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3567 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3568 root     15u  inet    4976        TCP *:www (LISTEN)
httpd      3569 root     15u  inet    4976        TCP *:www (LISTEN)
smbd       3591 root     5u   inet    5024        TCP *:netbios-ssn (LISTEN)
nmbd       3601 root     5u   inet    5046        UDP *:netbios-ns
nmbd       3601 root     6u   inet    5048        UDP *:netbios-dgm
nmbd       3601 root     8u   inet    5052        UDP localhost:netbios-ns
nmbd       3601 root     10u  inet    5054        UDP localhost:netbios-dgm
```

To reduce this list to just the names of the daemons themselves:

```
[root]# lsof -i -Fc | grep '^c' | cut -b2-20 | sort -u
httpd
lpd
named
nmbd
portmap
rpc.mountd
rpc.rquotad
rpc.statd
sendmail
smbd
```

Step 3.3.2. Eliminate unnecessary services

The trick is to know which init script controls which daemon. Most of the daemons are controlled by scripts of the same name, the others can be found by using grep to search for the name in `/etc/rc.d/init.d/*`. To eliminate the service, first shut it down with its init script, then remove the script from the runlevel directory. For Red Hat Linux, the easiest way to remove the links from the runlevel directories is with the `/sbin/chkconfig` program, or you can simply delete them. By default, `chkconfig` adds or removes links to scripts in levels 3, 4, and 5. It does not delete the actual script in `/etc/rc.d/init.d`. If you are manipulating the runtime scripts manually, remember to always remove the startup script from both the runlevel 3 and runlevel 5 directories, so that changing the default runlevel will not unintentionally open network services you thought were turned off.



The following will shutdown all services listed in Step 3.3.1 above:

```
[root]# /etc/rc.d/init.d/httpd stop
[root]# /sbin/chkconfig httpd off
[root]# /etc/rc.d/init.d/lpd stop
[root]# /sbin/chkconfig lpd off
[root]# /etc/rc.d/init.d/named stop
[root]# /sbin/chkconfig named off
[root]# /etc/rc.d/init.d/netfs stop
[root]# /sbin/chkconfig netfs off
[root]# /etc/rc.d/init.d/smb stop
[root]# /sbin/chkconfig smb off
[root]# /etc/rc.d/init.d/nfs stop
[root]# /sbin/chkconfig nfs off
[root]# /etc/rc.d/init.d/portmap stop
[root]# /sbin/chkconfig portmap off
[root]# /etc/rc.d/init.d/sendmail stop
[root]# /sbin/chkconfig sendmail off
```

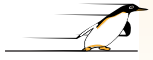
To remove the startup scripts manually, replace `chkconfig` with `rm` like so:

```
[root]# /bin/rm /etc/rc.d/rc[345].d/*sendmail
```

Once the service is turned off, consider removing the software package entirely. This frees up disk space and enhances security since local users can not abuse a service that isn't there.

Step 3.3.3. Check for any remaining services

Return to Step 3.3.1 and repeat with the `netstat` and `lsof` commands, checking for any remaining network services, and disable them until all network services are turned off.



STEP 3.4 GET THE LATEST VERSIONS OF SOFTWARE

In the remaining steps below, we discuss increasing the security of selected services. You should only enable the absolute minimum set of services necessary to do your work. Before enabling any services, make sure you have the latest version of the software. All the Linux vendors do a good job of keeping up with security-related bug fixes and update their FTP sites regularly with the latest software versions.

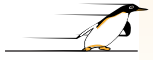
Step 3.4.1. Find security-related updates

Updated software, or "errata" in Red Hat's parlance, can be found on their web site at `http://www.redhat.com/corp/support/errata`, see Appendix A for the locations of updates for other distributions. Red Hat has recently revamped the errata page for Red Hat 6.0, with security-related updates in one column and other package updates in another. At a minimum, you should review and retrieve all of the security-related updates.

Step 3.4.2. Download updates

FTP URLs to the updated packages are included on the errata page itself, so downloading the packages can be done straight through your Web browser. A good practice is to create a separate directory, for example `/usr/local/updates`, and place the updated packages there. Updated packages can also be downloaded en masse from `ftp://updates.redhat.com`. Select the directory for the proper version number and architecture, e.g. `6.0/i386`, and download all the files into `/usr/local/updates`. (This can get tedious week after week, see Step 3.4.4 for information about automating the process).

RPM has the ability to install packages directly over the network by supplying a URL for the package name; this eliminates the need for a separate download directory. For workstations on slow and/or unreliable network connections, like modem lines, it is best to download the package and install from a local copy, rather than having the network link drop or hang halfway into an update.



Step 3.4.3. Install updates

Instructions for installing the updates with rpm are included in the errata pages. Generally, the package can be updated like so:

```
[root]# rpm -Uvh <package-name>
```

where `<package-name>` is the fully-qualified file name for the RPM package, and the options are: "U" for upgrade, "v" for verbose, "h" to print hash marks while it installs ("v" and "h" are for a nicer display, only "U" is required).

For example to update Samba:

```
[root]# rpm -Uvh samba-2.0.5a-1.i386.rpm
```

To update directly over the network, give a URL for the package name:

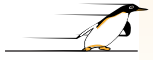
```
[root]# rpm -Uvh ftp://updates.redhat.com/6.0/i386/samba-2.0.5a-1.i386.rpm
```

If you have downloaded all of the latest updates packages into /usr/local/updates, you can "freshen" the packages that are already installed on your system with the command:

```
[root]# rpm -F /usr/local/updates/*
```

This will upgrade only those packages in `/usr/local/updates` that have already been installed, but with an earlier version number.

Note that to update the kernel, use `-i` for "install" rather than `-U` for "upgrade". This will leave the previous version of the kernel intact (the "U" option removes the previous kernel), so if the new kernel does not work for some reason, the machine can still be booted with the old kernel. Also, if you are running a modular kernel, especially on a SCSI-based system, you may need to rerun `mkinitrd` before booting the updated kernel. See the manual page for `mkinitrd(8)` for more information.



Step 3.4.4. Automate the process

It is important to update the packages before using the workstation for day-to-day work, but it is just as critical to stay up to date as time goes on. New bug fixes and enhancements are generated all the time. Trying to keep your local workstation synchronized with the updates can be tedious and time consuming.

Happily, there are several third-party tools that automate the process of checking an FTP site for new packages, and even installing the updates without human intervention. You can find these tools on the Metalab FTP site in the `/utils/package/` subdirectory, on Freshmeat, and many other popular Linux FTP sites. See Appendix A for specific URLs. We will discuss only one program here, but there are several others.

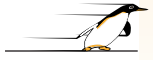
Step 3.4.4.1. Use AutoRPM to automate updates

See Appendix A for the URL of the AutoRPM home page. To quote from the AutoRPM man page:

```
AutoRPM is a program that can do any combination of the
following: mirror RPMs from an FTP site, keep installed
RPMs consistent with an FTP site or local directory, and
keep installed RPMs in a cluster or network of systems
consistent.  It is highly flexible.
```

To run AutoRPM you must first install the "`libnet`" Perl library, available from any Perl CPAN site. If you are unfamiliar with installing Perl CPAN libraries, the author of AutoRPM has thoughtfully provided an RPM version available from his Web site.

In our testing of AutoRPM for this guide, we found that the program is indeed very flexible, but a little tricky to install. The tar-ball could use a Makefile to make installation easier. The default configuration seems to work quite well for Red Hat Linux, and after a little browsing through the man pages, changing the configuration to find updates for other distributions was not difficult.



To install AutoRPM, download the compressed tar file and extract the program:

```
[root]# cd /usr/local/src
[root]# tar xzf <download-dir>/autorpm*.tar.gz
[root]# cd autorpm*
```

where `<download-dir>` is where the tar file was downloaded to. The wildcards in the commands simply avoid typing in the actual version number (which probably has changed by the time this guide went to press).

Create the configuration directory and install the configuration files, programs and man pages:

```
[root]# mkdir --mode 700 /etc/autorpm.d
[root]# cp autorpm.conf autorpm.d/* /etc/autorpm.d
[root]# mkdir --mode 700 /etc/autorpm.d/pools
[root]# cp pools/* /etc/autorpm.d/pools
[root]# install --owner=root --group=bin --mode=0700 autorpm.pl /usr/sbin/autorpm
[root]# cp autorpm.8 /usr/local/man/man8
[root]# cp autorpm.conf.5 /usr/local/man/man5
```

Read the manual pages for `autorpm(8)` and `autorpm.conf(5)`, but the default settings should work for Red Hat Linux.

Run AutoRPM manually in interactive mode to check for updated `RPMs`:

```
[root]# autorpm --interactive
```

AutoRPM will download the package names from the FTP sites listed in `/etc/autorpm.d/pools/redhat-updates` and compare them to the set of currently installed packages. The default list of FTP sites is quite long. It is good practice to edit the list and pare it down to only those sites you know and trust, such as `ftp://updates.redhat.com` and perhaps a local mirror. Next, an interactive dialog screen allows you to choose to update now, update later, or view RPM information such as files, dependencies, scripts, and signatures. The manual page has all the details.

Before you try to run AutoRPM to automatically download and install updates, edit `/etc/autorpm.d/redhat-updates` and add the command "`PGP_Require Yes`" in the action block labeled "`action (updated)`". This will force a PGP signature check of the RPM package before it is installed. You will need to install PGP or GPG and download the public keys for Red Hat before this will work. See the "PGP SIGNATURES" section of the `rpm(8)` man page for more information.

AutoRPM also comes with a cron job that can be placed in `/etc/cron.daily` or `/etc/cron.weekly`, whichever you prefer:

```
[root]# cp autorpm.cron /etc/cron.weekly/autorpm
```

The cron job is set up to mail reports of what needs to be done to the address listed in `/etc/autorpm.d/autorpm.conf` in the variable `ReportDest` (root, by default).

Configuring AutoRPM to work for other distributions can be as simple as changing the list of FTP sites in the `/etc/autorpm.d/pools/redhat-updates` file, however you may want to change the name to match your distribution. If so, you will need to edit `/etc/autorpm.d/autorpm.conf` and change the entry for `Config_File` to reflect the new name.

Step 3.4.5. Subscribe to security-related mailing lists

There are a lot of mailing lists that distribute information about computer security, newly discovered vulnerabilities and exploits, bug fixes, and updates. Even with an automated process for finding updated RPMs, it is important to stay informed about security-related problems for which updated software may not be available yet. At least you will know to turn a vulnerable service off until a fix is available.

See Appendix A for a list of security-related mailing lists and instructions for subscribing. For a typical workstation user, a high-volume mailing list like Bugtraq could result in information overload. At a minimum, subscribe to the "announce" list for your Linux distribution.

STEP 3.5 CACHING-ONLY DOMAIN NAME SERVICE (DNS)

Red Hat 6.0 includes the latest version of the Berkeley Internet Name Domain (BIND) software, version 8.2. This version includes several security-related bug fixes and additional security enhancements. Regardless, a Linux workstation has no need to run its own name server, even in a caching-only mode. Permanently disable and remove the software.

Step 3.5.1. Disable and remove DNS server software

Stop the `named` daemon, if it hasn't been done in step 3.3.2 above, and remove the software from the workstation:

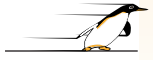
```
[root]# /etc/rc.d/init.d/named stop
[root]# rpm -e caching-nameserver
[root]# rpm -e bind
```

Step 3.5.2. Set primary and secondary name servers

If this was not done during the installation process, edit `/etc/resolv.conf` and add the IP addresses of your primary and, if available, secondary nameservers. Each of the name servers is denoted on a line by itself that looks like:

```
nameserver www.xxx.yyy.zzz
```

Logically enough, the IP for the primary nameserver is listed first, and the secondary nameserver second. Since the caching Domain Name Service has been removed, make sure that there is no `nameserver` line that points to your local IP number or 127.0.0.1.



A typical workstation does not need to run a Mail Transfer Agent (MTA), like `sendmail`, in daemon mode. Most Internet service providers, university, government, and commercial organizations provide mail servers.

Step 3.6.1. Turn off sendmail daemon mode

Edit `/etc/sysconfig/sendmail` to read:

```
DAEMON=no
QUEUE=15m
```

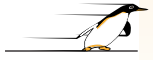
The QUEUE option controls how often sendmail will "wake up" to process the outgoing mail queue, which is set at 15 minutes in the above example. After saving the edited `/etc/sysconfig/sendmail`, restart the `sendmail` daemon:

```
[root]# /etc/rc.d/init.d/sendmail restart
```

Step 3.6.2. Define SMTP server for mail clients

Red Hat, like most Linux distributions, comes with a large number of Mail User Agents (MUAs), including mail (aka Mail), elm, mutt, pine, nmh, exmh, and Netscape. Each of these programs can be configured to send mail through an external MTA. Depending on the requirements for the organization, the external MTA will handle relaying the mail to its final destination and address translation.

The mail program (`/bin/mail`, aka `/usr/bin/Mail`), elm, mutt, and pine will all invoke `sendmail` stand-alone to deliver the mail to the given e-mail address, thus their configuration does not include an entry for an SMTP server. The assumption here, though, is that your organization has an SMTP server that handles relaying all out-bound electronic mail.



▲ **Step 3.6.2.1. Set out-bound SMTP server for sendmail**

You direct `sendmail` to relay all out-bound mail to the SMTP server with the `SMART_HOST` feature. There are two ways to do this, using the `m4` macros, or editing `/etc/sendmail.cf` directly. Using `m4` macros for anything other than the simplest modifications to `/etc/sendmail.cf` is the preferred method, and is discussed in Step 4.4 below. For now, the simplest way to set the out-bound SMTP server for sendmail is to edit `/etc/sendmail.cf` directly. Look for a line that starts with `DS` (this should be near the top of the file, just after a heading that says "`local info`"). Edit the line to read:

```
DSsmtp.my.domain
```

where `smtp.my.domain` is the fully qualified domain name of the SMTP server. To cover all bases for locally delivered mail, too, search for the lines that start with the two-letter combinations "`DR`", "`DH`", and "`DM`" edit them accordingly:

```
DRsmtp.my.domain
DHsmtp.my.domain
DMmy.domain
```

This will handle unqualified addresses (those without any @domain portion), local addresses (user@host without any other domain information), and sets the masquerade address, respectively. Save `/etc/sendmail.cf` and restart sendmail.

▲ **Step 3.6.2.1. Set out-bound SMTP server for other mail clients**

As mentioned above, Pine will default to using sendmail on the localhost for sending mail. You can set the SMTP server directly in Pine from the main menu by selecting "S" for setup, then "C" for config, moving down to the field labeled `smtp-server` and entering the hostname of the SMTP server.

The "new MH message system," nmh, and its X Window GUI front-end, exmh, can be configured to use an external SMTP server by editing the file `/etc/nmh/mts.conf`. Edit the `servers:` option, replacing the default "localhost" with the hostname of your SMTP server.

The SMTP server is set in Netscape Messenger by selecting the `Edit->Preferences...` dialog, selecting `Mail & Newsgroups->Mail Servers`, and placing the hostname of the SMTP server in the window under `Outgoing Mail Server`.

A typical workstation does not need to export directories to be NFS-mounted by another computer. The Network File System has inherent security problems because the Remote Procedure Call (RPC) mechanism it relies on uses IP number and UID for authentication, opening NFS to IP spoofing attacks.

Step 3.7.1. Turn off NFS exports and remove NFS daemons

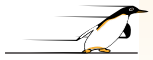
If there is no alternative to exporting NFS mount points, see Step 4.6 below for information on limiting access and permissions, otherwise turn off NFS and remove the NFS software completely:

```
[root]# /etc/rc.d/init.d/nfs stop
Shutting down NFS services:                [  OK  ]
Shutting down NFS mountd:                  [  OK  ]
Shutting down NFS daemon:                  [  OK  ]
Shutting down NFS quotas:                  [  OK  ]
Shutting down NFS statd:                   [  OK  ]
[root]# rpm -e knfsd
```

Step 3.7.2. Configure local NFS mounts

Even with the NFS daemons deleted from the workstation, you can still mount NFS directories from a central server as long as the NFS file system is compiled into the kernel. In fact for a workstation environment, this is the normal use of NFS. For example, say you are a member of a software development team and the source code for your project is maintained on a central server called `project` in the directory `/usr/src/devel`. You wish to mount this directory on your workstation on the directory `/mnt/devel`. Since all you need to do is copy source files in and out, mount the directory with the following options:

`rw`	Mount read-write
`hard`	Do a “hard” mount, i.e. hang if the server does not respond
`noexec`	Do not allow executing binaries
`nosuid`	Do not honor set-uid or set-gid bits (redundant with noexec)
`nodev`	Do not interpret character or block special files



For example:

```
[root]# mkdir /mnt/devel
[root]# mount -t nfs -o rw,hard,noexec,nosuid,nodev project:/usr/src/devel /mnt/devel
```

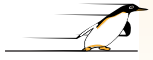
Note that read and write permissions on the NFS-mounted directory are defined by the UID and GID of the owner on the server (project), not the client (your workstation), so the UIDs and GIDs need to match up across both machines.

To have this directory mounted every time your workstation is booted, put the mount point in `/etc/fstab` like so:

```
/mnt/devel project:/usr/src/devel nfs rw,hard,noexec,nosuid,nodev,bg,auto 0 0
```

Adding `auto` allows for the init scripts to mount the NFS directory at boot-time (i.e. with `mount -a`). Note, however, that specifying a `hard` mount means that the workstation will hang if the NFS server is not up and running. Therefore, we have added `bg`, telling `/bin/mount` to do the mount in the background and allow us to proceed in the event the server is not immediately available. You may also wish to add another option, `intr`, to allow the mount process to be interrupted in case of a hang. If the NFS server or the local network have reliability problems, consider using a `soft` mount, as opposed to a `hard` mount. The advantage is that the workstation will not hang when the NFS server is down or unavailable; the disadvantage is that you may lose data on the NFS server if it is down and the client times out.

In the case that the NFS mount is expressly for the purpose of sharing executable programs (like `/usr/local/bin` on the NFS server, for example), you will need to change `noexec` to `exec`, but think hard before changing `nosuid` to `suid` to allow set-uid and set-gid permissions. Note that the `nosuid` and `noexec` options can still be defeated by programs like `suidperl` that copy a script as a normal data file, then execute it with SUID privileges. See the manual page for `mount(8)` and `fstab(5)` for more information.



STEP 3.8 LIMIT WORLD WIDE WEB SERVICES TO THE LOCAL HOST

Like most distributions, Red Hat 6.0 installs a fully functional Apache Web server that, by default, allows connections from any computer on the Internet. For a typical workstation setup, this is not necessary. If you are a Web developer, though, having a fully functional localized Web server is extremely useful for testing pages, CGI scripts, applets, etc. before they are deployed on an operational Web site.

Step 3.8.1. Turn off HTTP and remove the server software

If there is no need for a local web server, simply remove it from the workstation:

```
[root]# /etc/rc.d/init.d/httpd stop
Shutting down http:                        [   OK   ]
[root]# rpm -e apache
[root]# rpm -e apache-devel
```

Step 3.8.2. Limit HTTP access to localhost only

The Apache HTTP daemon can be bound to a specific network interface. If the interface is `localhost`, IP 127.0.0.1, then any attempt to connect to port 80 from an outside network results in a connection refused error. From the outside, it looks like there isn't any Web server on the machine at all.

Edit `/etc/httpd/conf/httpd.conf`, and look for a section near the bottom with the option "`Listen`" — it should come just before the section for `VirtualHost`. Add the following line:

```
Listen 127.0.0.1:80
```

Save `/etc/httpd/httpd.conf` and restart the Web server:

```
[root]# /etc/rc.d/init.d/httpd restart
Shutting down http:                        [   OK   ]
Starting httpd:                            [   OK   ]
[root]# netstat -a | grep www
tcp         0       0 localhost:www         *:*                 LISTEN
```

The `netstat` command shows us that the Web daemon is listening only on the loopback device, not the regular network device. To test that access is denied, from another computer, try:

```
$ telnet workstation 80
Trying 192.168.1.3...
telnet: Unable to connect to remote host: Connection refused
```

The Web server is now closed off to everyone except users on the workstation itself.

STEP 3.9 REMOVE ANONYMOUS FTP SERVICE

Even though you probably turned off the FTP service in Step 3.1.1 above, or limited access to it with TCP wrappers in Step 3.2.1, you should still eliminate the ability to do anonymous FTP. First edit `/etc/ftpaccess` and change the line that reads:

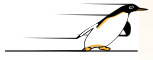
```
class    all    real,guest,anonymous    *
```

to read:

```
class    all    real   *
```

Finally, remove the home directory used for anonymous FTP:

```
[root]# rpm -e anonftp
```



STEP 4.1 SERVERS: SEE STEPS 3.1, 3.2, 3.3, AND 3.4 FOR DISABLING ALL UNNECESSARY SERVICES, SETTING WRAPPERS, AND UPDATING SOFTWARE

Like a workstation, a server should run only those few network services necessary to fulfill its purpose. An information server may be set up to run HTTP and FTP, but should not also bear the burden of operating large mailing lists or file sharing to a number of clients through Samba or NFS. While the remainder of this section provides steps for securing a number of services, this by no means implies that a single machine should provide all of them at the same time.

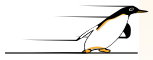
A typical server operates in a "lights-out" environment in a computer room. It should have as few user accounts as possible, and remote access to the server should be limited to Secure Shell only, to avoid the possibility of sniffing passwords on the network, and to provide strong authentication and encryption of the login session.

Consider whether any software development tools are necessary. A common mode of attack by crackers and "script kiddies" is to break into an ordinary user account, then download and compile exploit programs and root kits to compromise root and hide their tracks. If there is no C compiler or other development tools, it makes the task of the cracker that much harder. Development of tools and programs that have to run on the server, such as CGI scripts for a Web server, should be done on workstations, and tested thoroughly before installation on the operational server.

While you are removing the packages for the compilers and development libraries, you should also browse through the list of packages on the system with `rpm -qa` to see what other superfluous packages can be removed. In addition to the security benefits, removing unneeded packages frees up disk space that can be used for additional logging and temporary space. If you are not sure what a package is for, try `rpm -qi <package-name>` to get more information. In fact, it can be very useful to execute:

```
[root]# rpm -qail >/root/package.list
[root]# chmod 400 /root/package.list
```

This will dump a list of all installed packages, the package descriptions and a list of all the files in each package.



STEP 4.2 INSTALL SECURE SHELL FOR REMOTE ACCESS

Secure Shell (SSH) was originally developed at the Helsinki University of Technology. It provides strong authentication using RSA public key cryptographic algorithms and automatic and transparent encryption of network communications for login and file copying operations. Source code for version 1 of the SSH protocol is freely available for non-commercial use. The license for protocol version 2 source code is much more limited, and any use at a commercial or government organization is forbidden without a commercial license. See `http://www.ssh.fi/sshprotocols2/download.html` for links to the License Agreement for SSH2.

Step 4.2.1. Download, compile, and install SSH

SSH version 1 is available for download from `ftp://ftp.cs.hut.fi/pub/ssh/`, and if this site is too busy, there is a list of mirror sites on the SSH download page cited above. As of the writing of this guide `ssh-1.2.27.tar.gz` was the latest version, and we will use this version number in our examples below. Check the FTP site for later versions, since problems and bugs are corrected in the version 1 source code occasionally.

```
[root]# wget ftp://ftp.cs.hut.fi/pub/ssh/ssh-1.2.27.tar.gz
[root]# tar xzf ssh-1.2.27.tar.gz
[root]# cd ssh-1.2.27
[root]# ./configure --with-libwrap --without-rsh --disable-suid-ssh --disable-scp-stats
[root]# make
[root]# make install
```

The options to the `configure` script do the following: compile the SSH daemon with support for TCP wrappers; disables all support for using rsh as a fall-back option; disables the set-uid bit on the SSH client; and turns off the printing of on-screen statistics by the secure copy program, `scp` (to emulate the behavior of `rcp`).

Step 4.2.2. Start the SSH daemon

The SSH daemon, `sshd`, should be started at boot time and remain running as long as the server is up. A simple means for doing this is to add the line:

```
/usr/local/sbin/sshd
```

to `/etc/rc.d/rc.local`, where it will be executed after all other boot-time programs are started.

See Appendix C for a Red Hat-centric SYSV-init script using the `daemon` and `killproc` functions. Copy this script into `/etc/rc.d/init.d/sshd`, and link it into the runlevel directories:

```
[root]# for i in /etc/rc.d/rc[345].d; do (cd $i; ln -s ../init.d/sshd S50sshd); done
```

or, alternatively:

```
[root]# chkconfig sshd on
```

To start the daemon now, without rebooting:

```
[root]# /etc/rc.d/init.d/sshd start
```

Step 4.2.3. Set up `/etc/hosts.allow` for SSH access

Compiling SSH with the `–with-libwrap` configure option allows `sshd` to perform TCP-wrapper-style access control, inspecting the `/etc/hosts.allow` and `/etc/hosts.deny` files for each connection request. The syntax and options are exactly the same as any TCP-wrapped service, and the service name is `sshd`. For example:

```
# Allow SSH access to anyone in our domain or the goodguys.org domain
sshd: LOCAL .goodguys.org
```

Step 4.2.4. Generate SSH keys

SSH performs user authentication and the exchange of session encryption keys using RSA cryptographic public keys. Each user that is allowed to access the server must generate a public/private key pair on the workstation that they will use to access the server, and the public key must be placed on the server in the account that they wish to login to.

On your remote workstation, install SSH as per the instructions above, then in your regular user account do this:

```
[user]$ ssh-keygen
```

The key generator will compute the public and private key pair and prompt for the file to save them in, defaulting to `.ssh/identity` in your home directory. Accept the default location. Then you will be asked to enter a passphrase.

About passphrases: you are not required to enter one, and if you don't, SSH will not prompt you for a passphrase when it uses the key (like when you start a login session, for example). This is especially handy if you plan to automate `ssh` or `scp` (SSH secure copy) in scripts. The downside is that if your workstation and your account are compromised, the attacker will be able to access the server without entering a passphrase.

For ordinary shell and file copy access to the server, it is best to enter a passphrase, keeping in mind all the rules for selecting good passwords. If there is a need for automated, non-interactive access to the server using SSH, consider using `ssh-agent` and `ssh-add`. The agent is a process that holds private keys for use by `ssh` and `scp`. The `ssh-add` program communicates with `ssh-agent` process and sends it the key, prompting for the passphrase if necessary. After your key is stored by `ssh-agent`, you will not need to enter a passphrase again for the remainder of the login session. See the manual pages for `ssh-agent(1)` and `ssh-add(1)` for more information.

When `ssh-keygen` is done, there will be two files in the `.ssh` directory, `identity` and `identity.pub`. The former is your binary private key, the latter is an ASCII file with your public key. The public key has to be stored on the server before you can use SSH to login.

The safest way to transport the public key to the server is "sneaker-net", i.e., copy the key to a floppy and carry it to the server. Transmitting the public key over the network is generally safe, though. Even if the public key is sniffed off the network, it is useless without the private key and it is extremely difficult, if not impossible, to derive one key from the other. **Never** send the private key over the network, however, and make sure that the `.ssh/identity` file has 0600 permissions and is owned by you.

Once the key is copied to the server, put it in the home directory of the account you wish to access (we'll use the root account in our example) and assume the public key is in the file `identity.pub` in the home directory.

```
[root]# cd ~root
[root]# mkdir .ssh
[root]# cat identity.pub >>.ssh/authorized_keys
[root]# chmod 600 .ssh/authorized_keys
```

The `authorized_keys` file holds the public keys for every user that is granted access. If you copy more than one key using the method above, be careful with the redirection; make sure you append (>>) to the file, **do not** overwrite it (>).

Step 4.2.5. Use ssh and scp for remote access

With the SSH daemon running on the server, and the public key of a workstation user account in `/root/.ssh/authorized_keys`, you may now use SSH to access the server securely from your workstation.

The `ssh` command works just like `rsh`. To open a login shell:

```
[user@workstation]$ ssh -l root server
```

To execute a command:

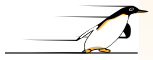
```
[user@workstation]$ ssh -l root server ls /home/httpd/html
```

The `scp` command works like `rcp`. To copy a file to the server:

```
[user@workstation]$ scp This.File root@server:That_File
```

To copy lots of files from the server:

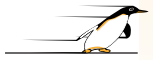
```
[user@workstation]$ scp root@server:All.* backup_dir
```



Note that if you entered a passphrase during the ssh-keygen stage, you will be prompted for that passphrase for each of these commands.

If `ssh` or `scp` do not work correctly, use the `-v` switch to see more detail about the interactions between the client and server:

```
[user@workstation]$ ssh -v -l root server
SSH Version 1.2.27 [i686-unknown-linux], protocol version 1.5.
Standard version.  Does not use RSAREF.
workstation: Reading configuration data /etc/ssh_config
workstation: ssh_connect: getuid 100 geteuid 0 anon 0
workstation: Connecting to server [192.168.1.3] port 22.
workstation: Allocated local port 1022.
workstation: Connection established.
workstation: Remote protocol version 1.5, remote software version 1.2.27
workstation: Waiting for server public key.
workstation: Received server public key (768 bits) and host key (1024 bits).
workstation: Host 'server' is known and matches the host key.
workstation: Initializing random; seed file /home/user/.ssh/random_seed
workstation: Encryption type: idea
workstation: Sent encrypted session key.
workstation: Installing crc compensation attack detector.
workstation: Received encrypted confirmation.
workstation: Trying rhosts or /etc/hosts.equiv with RSA host authentication.
workstation: Server refused our rhosts authentication or host key.
workstation: No agent.
workstation: Trying RSA authentication with key 'user@workstation'
workstation: Received RSA challenge from server.
workstation: Sending response to host key RSA challenge.
workstation: Remote: RSA authentication accepted.
workstation: RSA authentication accepted by server.
workstation: Requesting pty.
workstation: Requesting X11 forwarding with authentication spoofing.
workstation: Requesting shell.
workstation: Entering interactive session.
Last login: Sun Jul 25 11:08:51 1999 from workstation
You have new mail.
[root@server /root]#
```



Step 4.2.6. Replace "r" programs with SSH

Once SSH is installed and working, there is no need to have the "r" clients (`rlogin`, `rsh`, and `rcp`) or the server daemons (`in.rlogind`, `in.rshd`, or `in.rexecd`) on the workstation or the server at all. Even if they have been commented out of `/etc/inetd.conf`, removing the programs is the safest course.

```
[root]# rpm -e rsh
```

Once the "r" programs are gone, SSH programs can be put in their place:

```
[root]# ln -s /usr/local/bin/ssh /usr/bin/rsh
[root]# ln -s /usr/local/bin/slogin /usr/bin/rlogin
[root]# ln -s /usr/local/bin/scp /usr/bin/rcp
```

STEP 4.3 DOMAIN NAME SERVICE AND BIND VERSION 8

Red Hat 6.0 ships with BIND version 8.2, the latest version available as of this writing. Administrators of earlier releases of Red Hat and other distributions should check the version of BIND they are running. If you are running BIND Version 4, upgrade to BIND version 8.2. If you are running 8.1.1 or earlier you must upgrade to BIND Version 8.1.2 or later, because the earlier versions have a known remote root exploit. Check the updates for your distribution, or download the latest source from `http://www.isc.org`.

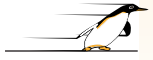
Step 4.3.1. Restrict zone transfers

Configure the primary master name server to perform zone transfers only to its slave servers, use the `allow-transfer` option for the zone in `/etc/named.conf`, and specify the IP address of only the slave servers for the zone:

```
zone "notransfer.com" {
        type master;
        file "db.notransfer.com";
        allow-transfer { 192.168.1.1; 192.168.2.1; };
};
```

The secondary, or slave, servers should not transfer zone information to any other name servers. In their copy of `/etc/named.conf`, put this:

```
zone "notransfer.com" {
        type slave;
        masters { 192.168.0.1; };
        allow-transfer { none; };
};
```



Step 4.3.2. Restrict queries

You can limit the hosts that are allowed to query a name server with the `allow-query` option. You can limit the interfaces that queries will be allowed on with the `listen-on` option. These options can be applied globally in the options list in `/etc/named.conf`:

```
options {
        allow-query { 192.168.0/24; };
        listen-on { 192.168.0.1; 127.0.0.1; };
};
```

Or the restriction can be for particular zones:

```
zone "notransfer.com" {
        type slave;
        masters { 192.168.0.1; };
        allow-transfer { none; };
        allow-query { 127/8; 192.168.0/24; };
};
```

This is useful for protecting zone information on a bastion host in a firewall environment. It allows queries only from the primary master nameserver on the internal network while a shadow primary master nameserver on the external network handles queries from the Internet and serves up a much more restricted view of the firewall network. See “DNS and BIND,” 3rd Edition, Ablitz and Liu, O’Reilly and Associates, 1998 for more information (this book is indispensable for any nameserver administrator).

Step 4.3.3. Run `named` in a chroot jail

BIND Versions 8.1.2 and 8.2 have the ability to run as a user other than root, which helps, but still leaves the system open if the nameserver is compromised. For better security, the service can be set up to run in its own chroot directory tree. Setting up a proper chroot jail must be done with care. Not only do the zone files and other configuration files need to be in the chroot directory tree, but libraries, devices, and executables necessary for operation of `named` have to be there, too. In the following example, the username (dns), UID, GID, and home directory were chosen arbitrarily. Use whatever unique values make sense for your site.

▲ **Step 4.3.3.1. Create the new user and group**

```
[root]#    groupadd -g 150 dns
[root]#    useradd -u 150 -g 150 -M dns
```

▲ **Step 4.3.3.2. Prepare the chroot directory**

```
[root]#    mkdir -m 0700 /home/dns
[root]#    cd /home/dns
[root]#    mkdir -p etc lib dev usr/sbin var/named var/run
[root]#    mknod -m 666 dev/null c 1 3
```

▲ **Step 4.3.3.3. Copy configuration files and programs**

```
[root]#    cp /etc/named.conf etc
[root]#    cp -R /var/named/* var/named
[root]#    chown -R dns.dns var/named var/run
[root]#    cp /usr/sbin/{named,named-xfer} usr/sbin
```

▲ **Step 4.3.3.4. Copy shared libraries**

First, figure out which libraries are needed:

```
[root]#    ldd /usr/sbin/named
           libc.so.6 => /lib/libc.so.6 (0x4001c000)
           /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
[root]#    cp /lib/libc.so.6 lib
[root]#    cp /lib/ld-linux.so.2 lib
```

▲ **Step 4.3.3.5. Set `syslogd` to listen to named logging**

`named` communicates with `syslog` through `/dev/log`, but this isn't possible in the chroot jail. The `syslog` daemon must be told to create a new socket for `named` to write to. Edit `/etc/rc.d/init.d/syslog`, and rewrite the line that starts the `syslog` daemon to read:

```
daemon syslog -a /home/dns/dev/log
```

And restart the daemon:

```
[root]# /etc/rc.d/init.d/named stop
[root]# /etc/rc.d/init.d/named start
```

▲ Step 4.3.3.6. Edit the named init script

Edit `/etc/rc.d/init.d/named` to start the daemon with the new UID/GID in the chroot jail:

```
start)
        # Start daemons.
        echo -n "Starting named: "
        daemon named -u dns -g dns -t /home/dns
        echo
        touch /var/lock/subsys/named
        ;;
```

And restart the nameserver:

```
[root]# /etc/rc.d/init.d/named stop
[root]# /etc/rc.d/init.d/named start
```

To be sure it worked, check `/var/log/messages`, `named` should have written entries that read something like:

```
Jul 25 01:08:00 server named[1782]: chrooted to /home/dns
Jul 25 01:08:00 server named[1782]: group = dns
Jul 25 01:08:00 server named[1782]: user = dns
```

Use `nslookup` to test that the nameserver is operating as expected.

▲ Step 4.3.3.7. Specify a new control channel for ndc

If you need to run the nameserver daemon control program, `ndc`, you will have to specify a new control channel within the chroot jail:

```
[root]# ndc -c /home/dns/var/run/ndc
```

A shell alias or a short script with the new control channel can be used to save keystrokes.

As mentioned in Step 3.6 above, `sendmail` is the Mail Transfer Agent (MTA) supplied with Red Hat 6.0 and most other distributions. Red Hat 6.0 ships with `sendmail` version 8.9.3, the latest version available as of this writing. Administrators of earlier releases of Red Hat or other distributions should upgrade to the version 8.9.3 of `sendmail`. The 8.9.x versions of `sendmail` disallow electronic mail relay by default. Mail relay is a favorite tool of the purveyors of "spam" bulk mail to use your server to forward unsolicited mail.

The configuration and operation of `sendmail` is quite complicated. All `sendmail` administrators should have a copy of "sendmail", Costales and Allman, O'Reilly & Associates, 1997 (the Bat Book) on their bookshelf. Modifying `/etc/sendmail.cf` by hand can be difficult and fraught with danger for the beginner. The best way to make changes is to use the `m4` macro processor system, package `sendmail-cf*.rpm` on the Red Hat distribution media. Full details on using `m4` can be found in the Bat Book.

Step 4.4.1. Turn off SMTP `vrfy` and `expn` commands in `/etc/sendmail.cf`

The `vrfy` SMTP command allows a remote user to verify the E-mail address of a local user on the server. The `expn` SMTP command expands aliases and mailing list `:include:` aliases. Local addresses and aliases should be considered confidential, or at least sensitive, information. To turn off the commands, edit `/etc/sendmail.cf` and search for the line with `PrivacyOptions`. Change it to read:

```
O PrivacyOptions=goaway
```

The `goaway` option is shorthand for `authwarnings,noexpn,novrfy,needmailhelo,needexpnhelo, needvrfyhelo`; complete descriptions of each of these privacy options can be found in the Bat Book.

To set this option with `m4` macros, add the following to the build macro (for Red Hat Linux, this is file `/usr/lib/sendmail-cf/cf/redhat.mc`) and rebuild `/etc/sendmail.cf`:

```
define(`confPRIVACY_FLAGS', `goaway')
```

A sample procedure for this follows:

```
[root]# cp /etc/sendmail.cf /etc/sendmail.cf.orig
[root]# cd /usr/lib/sendmail-cf/cf
[root]# echo "define(\`confPRIVACY_FLAGS', \`goaway')" >>redhat.mc
[root]# m4 redhat.mc >/etc/sendmail.cf
[root]# /etc/rc.d/init.d/sendmail stop
[root]# /etc/rd.c/init.d/sendmail start
```

Step 4.4.2. Define hosts allowed to relay mail

Sendmail V8.9 has built-in checks to stop mail relays. Using unwitting mail servers to relay unsolicited mail is a favorite practice of spammers. An organizational mail server has to allow mail relay for mail clients inside the organization, though. Relaying is controlled through an access database, which is in the file `/etc/mail/access` in Red Hat Linux. Other distributions may control relaying through the file `/etc/mail/relay-domains`. Both methods are discussed below.

Step 4.4.2.1. Check that the access database is active

To make sure `sendmail` has the ability to use the access database check for the name of the database in `/etc/sendmail.cf`:

```
[root]# grep Kaccess /etc/sendmail.cf
Kaccess hash -o /etc/mail/access
```

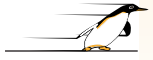
If the line isn't found, the only easy way to add this feature is with the m4 macro `FEATURE(`access_db').` See the sample procedure in Step 4.4.1 above.

If your distribution uses the `relay-domains` file, instead, check to make sure that `sendmail` is using the file:

```
[root]# grep relay-domains /etc/sendmail.cf
FR-o /etc/mail/relay-domains
```

If the line isn't found, edit `/etc/sendmail.cf` and add the line just as it appears above, create the `relay-domains` file and restart the `sendmail` daemon:

```
[root]# touch /etc/mail/relay-domains
[root]# chmod 600 /etc/mail/relay-domains
[root]# /etc/rc.d/init.d/smtp restart
```



▲ Step 4.4.2.2. Set access for domains allowed to relay

The access database has a simple "key value" format: the key is a fully qualified hostname, subdomain, domain, or network; the value is an action, REJECT, DISCARD, OK, RELAY, or an arbitrary message. Only hosts or domains with the RELAY action are allowed to use the mail server as a mail relay. For example, if your domain is example.org, and your server is also handling mail for all hosts on the 128.184.x.x network, edit `/etc/mail/access` to read:

```
example.org                RELAY
128.184                    RELAY
```

For distributions that use `relay-domains`, instead, the file `/etc/mail/relay-domains` should be added to include just those host and domain names that are allowed to relay mail through the server, and no others. Remember to restart the sendmail daemon after modifying the database.

Much more information about relaying and anti-spam configuration control can be found at `http://www.sendmail.org/m4/readme.html`.

■ Step 4.4.3. Set domain name masquerading

Many organizations prefer to have uniform mail addresses for all personnel, such as `Joe.Schmo@example.org`. Some mail clients can be set so that the uniform mail address is used on all out-bound mail, but managing all the mail clients in even a small group can be tedious. Sendmail can be configured to rewrite the headers of all out-bound mail so that they masquerade as the central mail server. To define the masquerade address, edit `/etc/sendmail.cf` and look for `DM` in columns 1-2. Append the masquerade name to this line:

```
DMexample.org
```

Or, to be complete, use the `m4` macros and add the following:

```
MASQUERADE_AS(`example.org')
FEATURE(masquerade_entire_domain)
FEATURE(allmasquerade)
FEATURE(masquerade_envelope)
```