Linux Secure and Optimized Server

A guide for information system, configuration, optimization and network security professionals.

50 Quintin suite 101
St-Laurent H4N 3A5
Quebec Canada
Mail: **gmourani@videotron.ca**
Author: Gerhard Mourani
Version: 1.0
Last Revised: November 1, 1999

Linux **S**ecure & **O**ptimized **S**erver

New version of this document

New version of this document will be periodically posted to http://pages.infinit.net/lotus1/doc/opti/Linuxsos.pdf. All comments, error reports, additional information, criticism and money of all sorts should be directed to gmourani@videotron.ca. If you send E-mail please make sure that the return address is correct and working, I get a lot of E-mail and figuring out your e-mail address can be a lot of work.

If you want to translate this documentation please notify me so I can keep track of what languages I have been published in.

Copyright Information

PGP Public Key for Gerhard Mourani

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.0.0 (GNU/Linux)
Comment: For info see http://www.gnupg.org

mQGiBDgU8UcRBADiuIKn95nz0qsvjU1GzBxv0AOxJHVTNhFBl6lt+3DzDA0G7UTu
hOhT0aGwVGts3bzjXVbhS44CTfAvvuVYQq7Ic/BHkwIhFvSu/Xv/fGbD3IQy+Gn5
UYzhZegCGwB0KQhGkIwQPus2ONOS5oT3ChZ8L7JlCPBnlOcVBT+hZ3BXUwCg4y4L
Mz5aEe0MPCZ3xkcNE7AE71EEAL4Jf2uVhIRgOfwIpdB1rKVKrDDFxZLx+yZeOZmq
gdwa4m7wV+Rk+c4I1+qBxxkmcUBhTHigx+9kpBDE2J0aEGQezDN+RoqlmdyVFO98
T/znf4ZLIf0upu5aP4kAItJJuFB1AaJyDLesB5xGjfyWz+RhbKOmeqr2zHniOsa8
HcZ/BACKZFBjNElqFUf0niWf822W6IbNf7ASh8pwTgR9PmXcq2qtBBq8uCIpEYcD
wzk+ccl2jt8qt5RB7DXz/r/uG+3YHU+ID4iz6Qm6zl84gYQLDXST2YXZ5BPURo7H
O4nEIJfeHEuUCstE5ROKnblG2U+t5QmxSGbETnK9I/OZrzFwILRDR2VyaGFyZCBN
b3VyYW5pIChPcGVuIE5ldHdvcmsgQXJjaGl0ZWN0dXJlKSA8Z21vdXJhbmlAdmlk
ZW90cm9uLmNhPohVBBMRAgAVBQI4FPFHAwsKAwMVAwIDFgIBAheAAAoJEDPaC2+7
tLqbGcYAnjHIPAsZrRC5qU5OrqdPvvEmICUWAKCdeyWwJ785A58U8Vh1bpxzCVVb
PbkCDQQ4FPI0EAgAy7qa88bVYWIEyAWxJPZRxl8G2GcxgshSu4+5udeP+4PlVAm8
3DUynzlcax4/ikx8Q8MoVR7s6lCLJXCycLENE8xFCJJQ26IxzBjdftGdmvKteVkZ
Kld9PZMzjUsxKzmhZbGEWug6xaav68EIewTw/S0TFtPhXyUKFrYPV6aID7YGatzB
P4hQJfh4Wt3NdP9QznASBze6bPZxR07iEZaUO0AMHeeBKwL6rptEcGuxHPMYc00R
s+SdGTOAa9E/REIiiEike9mXTKKWJYG2e7leDP3SBruM/c7n+DC9ptFAapg1GD9f
Re7LLFqj6EQzZqybPB61B9rB/8ShIrApcNYF4wADBQgAvROi9N0/J5kYvBVb60no
xBUBYtZp4cJO9X1uVdVahCb9XZpbvxhKujaUoWpPCIb0pm8K+J8x0o9HFl9f/JTs
25N/eJwksr63+j8OdCHqxv4z+qQYgc/qvU42ekHlSfMc7vsiAIE1e1liuTBdN9KR
7oSBoaht+dKi16ffxXmMDvQs1YSBR114XXDSzI+xXRuaIISpi75NE6suLLlrksnL
+i/NcLRbCTEv4p1UJGYT4OVnX6quC3CC+U4Drpjf2ohawsXqS7jKUYduZRr9Hbar
/sE0pQ/P0uf+VAspQJgpvBqiDxbIRCDSx8VgDoRL7iayxPDXtFmbPOrUEPdS7qYX
pIhGBBgRAgAGBQI4FPI0AAoJEDPaC2+7tLqbdzQAniStW48nFU6CWkvQTy8fr0lu
ZXmXAKC5bgSLgg1gZAvx61Z20yzM+hwNFQ==
=95nO

-----END PGP PUBLIC KEY BLOCK-----

Overview

This document is tailored as a step-by-step, example driven document instead of a detailed explanation document on each Linux feature. It doesn't go into much debugging aspects since the Linux Documentation Project's (LDP) HOWTOs already cover this. This document is intended for a technical audience! It's discuss how to install a RedHat Linux Server with all the necessary security and optimization for a high performance Linux specific machine. Since we speak of optimization and configuration options, we will use a source distribution (tar.gz) program the most possible especially for critical server software like Apache, Bind, Samba, Squid, Openssl etc. Source program will give us a fast upgrade when necessary and a customization, optimization for our specific machines that often we can't have with RPM. We've used many freely available sources to write this documentation, it seems only fair to give the work back to the Linux community. It is focused on the Intel x86 hardware, so if you are looking for PPC, ARM, SPARC, APX, etc., features; you probably won't find what you are looking for. Minimal installation for this Server require that you recompile the kernel and install DNS Server, other programs are specific according to your needs.

These installation instructions assume

You have a CD-ROM drive and the Official Red Hat Linux CD-ROM.
Installations were tested on the Official RedHat Linux 6.1.

You should understand the hardware system on which the operating system will be installed. After examining the hardware, the rest of this document guides you, step-by-step, though the installation process.

Know your Hardware!

Understanding the hardware is essential for a successful installation of RedHat Linux. Therefore, you should take a moment now and familiarize yourself with your hardware. Be prepared to answer the following questions:

1. How many hard drives do you have?
2. What size is each hard drive (3.2GB)?
3. If you have more than one hard drive, which is the primary one?
4. How much RAM do you have?
5. Do you have a SCSI adapter? If so, who made it and what model is it?
6. What type of mouse do you have?
7. How many buttons?
8. If you have a serial mouse, what COM port is it connected to?
9. What is the make and model of your video card? How much video RAM do you have?
10. What kind of monitor do you have (make and model)?
11. Will you be connecting to a network? If so, what will be the following:
    a. Your IP address?
    b. Your netmask?
    c. Your gateway address?
    d. Your domain name server's IP address?
    e. Your domain name?
    f. Your hostname?
    g. Your types of network(s) card(s) (make and model)?

I) Creating the Boot Disk and Booting

Before you make the boot disk, insert the Official Red Hat Linux 6.1 CD-ROM Part 1 in your computer. When the program asks for the filename, you enter **boot.img** for the boot disk. To make the floppies under MS-DOS, you need to use these commands (assuming your CD-ROM is drive D: and contain the Official Red Hat Linux 6.1 CD-ROM).

Open the Command Prompt under Windows: Start | Programs | Command Prompt
Type **d:**
Type **cd \images**
Type **\dosutils\rawrite.exe**
D:\images>\dosutils\rawrite.exe
Enter disk image source file name: **boot.img**
Enter target diskette drive: **a**

-rawrite.exe asks for the filename of the disk image. Enter **boot.img**. Insert a floppy into drive A. It will then ask for a disk to write to. Enter **a:**. Label the disk Red Hat boot disk.

Since we start the installation directly off the CD-ROM, you have to boot with the boot disk. Insert the boot disk you create into the drive A on the computer where you wan to install Linux and reboot the computer. At the boot: prompt, press "**Enter**" to continue booting.

Choose your language
Choose your keyboard type
Select **"**Local CD-ROM**"**
Select your mouse type

II) Installation Class and Method

RedHat Linux 6.1 includes defines four different classes, or type of installation. They are:

GNOME Workstation
KDE Workstation
Server
Custom

These classes (*GNOME Workstation, KDE Workstation, and Server*) give you the option of simplifying the installation process (with a lot loss of configuration flexibility that we don't want to have). For this reason we highly recommend "*Custom*", as this allows you to choose what services are added and how the system is partitioned. The idea is to load the minimum packages, while maintaining maximum efficiency. The less software that resides on the box, the fewer potential security exploits or holes.

Select "**Custom**" and click Next.

III) Disk Setup

Warning
We highly recommend, therefore, that you make a backup of your current system before proceeding with the disk partitioning.

For performance, stability and security reason you must do something like the following partition listed bellow on your system. We suppose for this partition configuration the fact that you wan to setup a Web server with a Proxy Server on your Server Machine. We will make two special

partitions (chroot and cache), "chroot" partition is for DNS server chrooted, Apache server chrooted and other chrooted future programs. The "cache" partition is for our Squid Proxy server. If you are not intended to install Squid Proxy server you don't need to create the "cache" partition but remember that Squid + Apache will improve a lot your machine performance and security. Other partitions are *"/var",* by isolating the *"/var"* partition, you protect your root partition from overfilling. *"/tmp*" for the same reason like *"/var*", *"/home*" for users and *"/usr*" by default. In our partition configuration we'll reserve 400 MB of disk space for chrooted programs like Apache, DNS and other. This is necessary because Apache DocumentRoot files and other binaries, programs related to Apache will be installed in this partition. Take a note that the size of the Apache chrooted directory on the chrooted partition is proportional to the size of your DocumentRoot files. If you're not intended to install and use Apache on your server, you can reduce the size of this partition to something like 10 MB for DNS server that you always need.

Disk Druid Partitions is a program that partition your hard drive for you. Choose "**Add**" to add new partition, "**Edit**" to edit partition, "**Delete**" to delete partition and "**Reset**" to reset partition to the original state. When adding a new partition, a new window appear on your screen and give you parameters to choose. Different parameters are:

Mount Point: for where you wan to mount you new partition.
Size (Megs): for the size of your new partition in megabyte.
Partition Type: Linux native for Linux fs and Swap for Linux Swap Partition.

If you have a SCSI disk the device will be ***/dev/sda*** and if you have an IDE disk it will be ***/dev/hda***. If you looking for high performance and stability, a SCSI disk is highly recommended.

Linux refers to disk partitions using a combination of letters and numbers. It's uses a naming scheme that is more flexible and conveys more information than the approach used by other operating systems. Here is a summary:

First Two Letters – The first two letters of the partition name indicate the type of device on which the partition resides. You'll normally see either **hd** (for IDE disks), or **sd** (for SCSI disks).

The Next Letter – This letter indicates which device the partition is on. For example, ***/dev/hda*** (the first IDE hard disk) and ***/dev/hdb*** (the second IDE disk).

Keep this information in mind, it will make things easier to understand when you're setting up the partitions Linux requires.

A swap partition – Swap partition are used to support virtual memory. If your computer has 16 MB of RAM or less, you must create a swap partition. Even if you have more memory, a swap partition is still recommended. The minimum size of your swap partition should be equal to your computer's RAM or 16 MB (whichever is larger). The largest useable swap partition is roughly 1 GB, (since 2.2 kernel, 1 GB swap file are supported) so making a swap partition larger than that will result in wasted space. Note, however, that you can create and use more than one swap partition (although this is usually only necessary for very large server installations).

Try to put your swap partitions near the beginning of your drive. The beginning of the drive is physically located on the outer portion of the cylinder, so the read/write head can cover much more ground per revolution.

Now for example:
To make the partitions listed bellow on your system (this is the partition we'll need for our server installation); the command will be under Disk Druid:

Add
Mount Point: **/boot** ← our /boot directory.

Size (Megs): **5**
Partition Type: **Linux Native**
Ok

Add
Mount Point: **/usr** ← our /usr directory.
Size (Megs): **1000**
Partition Type: **Linux Native**
Ok

Add
Mount Point: **/home** ← our /home directory.
Size (Megs): **500**
Partition Type: **Linux Native**
Ok

Add
Mount Point: **/chroot** ← our /chroot directory.
Size (Megs): **400**
Partition Type: **Linux Native**
Ok

Add
Mount Point: **/cache** ← our /cache directory.
Size (Megs): **400**
Partition Type: **Linux Native**
Ok

Add
Mount Point: **/var** ← our /var directory.
Size (Megs): **200**
Partition Type: **Linux Native**
Ok

Add
Mount Point: ← our /Swap partition (leave the Mount Point Blank).
Size (Megs): **150**
Partition Type: **Linux Swap**
Ok

Add
Mount Point: **/tmp** ← our /tmp directory.
Size (Megs): **100**
Partition Type: **Linux Native**
Ok

Add
Mount Point: **/** ← our / directory.
Size (Megs): **316**
Partition Type: **Linux Native**
Ok

You must see something like the following information on your screen. Our mount point will look like that:

Mount Point	**Device**	**Requested**	**Actual**	**Type**

/boot	Sda1	5M	5M Linux Native
/usr	Sda5	1000M	1000M Linux Native
/home	Sda6	500M	500M Linux Native
/chroot	Sda7	400M	400M Linux Native
/cache	Sda8	400M	400M Linux Native
/var	Sda9	200M	200M Linux Native
<Swap>	Sda10	150M	150M Linux Swap
/tmp	Sda11	100M	100M Linux Native
/	Sda12	316M	315M Linux Native

Drive	Geom [C/H/S]	Total (M)	Free (M)	Used (M)	Used (%)
sda	[3079/64/32]	3079M	1M	3078M	99%

Now that you are partitioning and choosing the mount point of your directories, select “Next” to continue. After your partitions are created, the installation program will ask you to choose partitions to format. Choose the partitions you want to initialize, check the **(Check for bad blocks during format)** box, and press “Next”. This formats the partitions and makes them active so Linux can use them.

On the next screen you will see the LILO Configuration where you have the choice to install LILO boot record on:

Master Boot Record (MBR)
Or
First Sector of Boot Partition

Usually if Linux is the only OS on your machine you must choose “Master Boot Record (MBR)”. After you need to configure your LAN and clock. After you finish configuring the clock, you need to give your system a root password and authentication configuration.

NOTE: For Authentication Configuration don’t forget to select: 1) Enable MD5 passwords and 2) Enable Shadow passwords. Enable NIS doesn’t need to be selected since we are not configuring a NIS service on this server.

IV) Components to Install

After your partitions have been configured and selected for formatting, you are ready to select packages for installation. By default, Linux is a powerful operating system that executes many useful services. However, most of these services are unneeded and pose a potential security risk. You first have to choose which system components you want to install. Choose the components, and then you can go through and select or deselect each individual package of each component by selecting (**Select individual packages**) option. Since we are configuring a Linux Server, we don’t need to install a graphical interface (XFree86) on our system (graphical interface on a server mean; less process; less cpu; less memory; security risks and so on). Graphical interface is usually used on workstation only. Select the following packages for installation.

Networked Workstation
Network Management Workstation
Utilities

After selecting the components you wish to install, you may select or deselect packages.

Note: select the (**Select individual packages)** options (very important) before continuing to have the possibility to select and deselect packages.

Individual Packages Selection

The installation program presents a list of the package groups available, select a group to examine.

Component listed bellow must be unselected from the Menu Group for security; optimization and other reason described bellow.

Applications/File:	git
Applications/Internet:	finger, ftp, fwhois, ncftp, rsh, rsync, talk, telnet
Applications/Publishing:	ghostscript, ghostscript-fonts, mpage, rhs-printfilters
Applications/System:	arpwatch, bind-utils, knfsd-clients, rdate, rdist, screen, ucd-snmp-utils
Documentation	indexhtml
System Environment/Base:	chkfontpath, ipchains, yptools
System Environment/Daemons:	XFree86-xfs, knfsd, lpr, portmap, routed, rusers, rwho, tftp, ucd-snmp, ypbind
System Environment/Libraries:	XFree86-libs, libpng
User Interface/X:	XFree86-75dpi-fonts, urw-fonts

Before we explain each description of programs we wan to uninstall, someone can ask why I need to uninstall finger, ftp, fwhois and telnet on the server? First of all we know that those programs by their nature are insecure. Now imagine that cracker have acceded your new server, he can use finger, ftp, fwhois and telnet programs to query or access other node on your network. If those programs are not installed on your new server, he will be compelled to use those programs from the outside or try to install program on your server in which case you can trace it with toll like Tripwire.

Finger is a utility, which allows users to see information about system users (login name, home directory, name, how long they've been logged in to the system, etc.). **[Security risks]**

The ftp package provides the standard UNIX command-line FTP client. FTP is the file transfer protocol, which is a widely used Internet protocol for transferring files and for archiving files. **[Security risks]**

The fwhois program allows you or your system's users for querying whois databases. **[Security risks]**

Ncftp is an improved FTP client. Ncftp's improvements include support for command line editing, command histories, recursive gets, automatic anonymous logins and more. **[Security risks, unnecessary]**

The rsh package contains a set of programs, which allow users to run commands on remote machines, login to other machines and copy files between machines (rsh, rlogin and rcp). **[Security risks]**

The ntalk package provides client and daemon programs for the Internet talk protocol, which allows you to chat with other users on different systems. **[Security risks]**

Telnet is a popular protocol for logging into remote systems over the Internet. **[Security risks]**

Ghostscript is a set of software that provides a PostScript(TM) interpreter, a set of C procedures (the Ghostscript library, which implements the graphics capabilities in the PostScript language) and an interpreter for Portable Document Format (PDF) files. **[Unnecessary]**

These fonts can be used by the GhostScript interpreter during text rendering. They are in addition to the shared fonts between GhostScript and X11. **[Unnecessary]**

The mpage utility takes plain text files or PostScript(TM) documents as input, reduces the size of the text, and prints the files on a PostScript printer with several pages on each sheet of paper. **[Unnecessary and not printer installed on the server]**

The rhs-printfilters package contains a set of print filters, which are primarily meant to be used with the Red Hat printtool. **[Unnecessary and not printer installed on the server]**

Texinfo is a documentation system that can produce both online information and printed output from a single source file. **[Unnecessary]**

Bind-utils contains a collection of utilities for querying DNS (Domain Name Service) name servers to find out information about Internet hosts. **[We will compile it later on this documentation]**.

The rdate utility retrieves the date and time from another machine on your network, using the protocol described in RFC 868. **[Security risks]**

The rdist program maintains identical copies of files on multiple hosts. If possible, rdist will preserve the owner, group, mode and mtime of files and it can update programs that are executing. **[Security risks]**

The ucd-snmp package contains various utilities for use with the UCD-SNMP network management project. **[Unnecessary, Security risks]**

The indexhtml package contains the HTML page and graphics for a welcome page shown by your Web browser, which you'll see after you've successfully installed Red Hat Linux. **[Unnecessary]**

Chkfontpath is a simple terminal mode program for adding, removing and listing the directories contained in the X font server's path. **[Unnecessary]**

Linux IP Firewalling Chains is an update to (and hopefully an improvement upon) the normal Linux Firewalling codes, for 2.0 and 2.1 kernels. It let's you do things like firewalls, IP masquerading, etc. **[Unnecessary, not a firewall server]**

The Network Information Service (NIS) is a system, which provides network information (login names, passwords, home directories, group information) to all of the machines on a network. **[Security risks]**

The knfsd is the *new* kernel NFS server and related tools. It provides a much higher level of performance than the traditional Linux user-land NFS server. **[Security risks]**

The lpr package provides the basic system utility for managing printing services. **[Unnecessary and not printer installed on the server]**

The portmapper program is a security tool which prevents theft of NIS (YP), NFS and other sensitive information via the portmapper. A portmapper manages RPC connections, which are used by protocols like NFS and NIS. **[Unnecessary, Security risks]**

The routed routing daemon handles incoming RIP traffic and broadcasts outgoing RIP traffic about network traffic routes, in order to maintain current routing tables. These routing tables are essential for a networked computer, so that it knows where packets need to be sent. **[Unnecessary, Security risks]**

The rwho command displays output similar to the output of the who command (it shows who is logged in) for all machines on the local network running the rwho daemon. **[Security risks]**

The Trivial File Transfer Protocol (TFTP) is normally used only for booting diskless workstations. The tftp package provides the user interface for TFTP, which allows users to transfer files to and from a remote machine. **[Security risks, Unnecessary]**

SNMP (Simple Network Management Protocol) is a protocol used for network management (hence the name). **[Unnecessary, Security risks]**

This is a font server for XFree86. You can serve fonts to other X servers remotely with this package, and the remote system will be able to use all fonts installed on the font server, even if they are not installed on the remote computer. **[Unnecessary]**

XFree86-libs contain the shared libraries that most X programs need to run properly. These shared libraries are in a separate package in order to reduce the disk space needed to run X applications on a machine without an X server (i.e, over a network). **[Unnecessary]**

Free versions of the 35 standard PostScript fonts. With newer releases of ghostscript quality versions of the standard 35 Type 1 PostScript fonts are shipped. **[Unnecessary]**

XFree86-75dpi-fonts contain the 75 dpi fonts used on most X Window Systems. **[Unnecessary]**

At this point, the installation program will format every partition you selected for formatting. This can take several minutes depending of the speed of your machine. Once all partitions have been formatted, the installation program starts to install packages.

V) How to use RPM Commands

This section contains an overview of principal modes using with RPM for installing, uninstalling, upgrading and querying RPM packages on your Linux system.

To install a RPM package, use the command:
[root@deep]# **rpm -ivh foo-1.0-2.i386.rpm**

RPM packages have file names like **foo-1.0-2.i386.rpm**, which includes the package name (**foo**), version (**1.0**), release (**2**), and architecture (**i386**).

To uninstall a RPM package, use the command:
[root@deep]# **rpm -e foo**

Notice that we used the package name "**foo**", not the name of the original package file "**foo-1.0-2.i386.rpm**".

To upgrade a RPM package, use the command:
[root@deep]# **rpm -Uvh foo-1.0-2.i386.rpm**

RPM automatically uninstall the old version of foo package and install the new one. Always use –Uvh to install packages, since it works fine even when there are no previous versions of the package installed.

To query a RPM package, use the command:
[root@deep]# **rpm -q foo**

This command will print the package name, version, and release number of installed package foo. Use this command to verify if package are or are not installed on your system.

VI) Starting and stopping daemon services

Init is the program that gets run by the kernel at boot time. It is in charge of starting all the normal processes that need to run at boot time. These include the APACHE daemons, NETWORK daemons, and anything else you want to run when your machine boots.

How does init start and stop services? Each of the scripts is written to accept an argument, which can be “start” and “stop” and are located under */etc/rc.d/init.d/* directory. You can execute those scripts by hand in fact with a command like:

For example:
To start the httpd Web Server under Linux.
[root@deep]# **/etc/rc.d/init.d/httpd start**

To stop httpd Web Server under Linux.
[root@deep]# **/etc/rc.d/init.d/httpd stop**

Check inside your */etc/rc.d/init.d/* directory for services available and use command start | stop to work around.

VII) Software that must be uninstalled after installation of the Server

Red Hat Linux install other pre-established program in your system by default and don’t give you the choice to uninstall them during the install setup. For this reason, you must uninstall the following software on your system after the installation of your server:

pump, mt-st, eject, bc, mailcap, quota, apmd, kernel-pcmcia-cs, linuxconf, getty_ps, setconsole, isapnptools setserial XFree86-video-card kudzu raidtools and gnupg.

Use command RPM like the following to uninstall them. Command to uninstall software is:

[root@deep]# **rpm -e softwarenames**

[root@deep]# **rpm -e pump mt-st eject bc mailcap quota apmd kernel-pcmcia-cs linuxconf getty_ps setconsole isapnptools setserial XFree86-video-card kudzu raidtools gnupg**
[root@deep]# rm -f /etc/conf.linuxconf-installed (this is a configuration file related to Linuxconf that must be removed manually).

hdparm is needed by IDE disk but not SCSI disk. If you are an IDE disk you must keep this program (hdparm) and if you don’t have an IDE disk you can remove it from your system.

Program **sendmail**, **procmail** and **mailx** are needed if you wan to configure your server as mail server. If your are not intend to use your new server as mail server you can uninstall them for security reasons.

[root@deep]# **rpm -e sendmail procmail mailx**

Pump DHCP (Dynamic Host Configuration Protocol) and BOOTP (Boot Protocol) are protocols which allow individual devices on an IP network to get their own network configuration information (IP address, subnetmask, broadcast address, etc.) from network servers. **[Unnecessary]**

The mt-st package contains the mt and st tape drive management programs. Mt (for magnetic tape drives) and st (for SCSI tape devices) can control rewinding, ejecting, skipping files and blocks and more. **[Unnecessary]**

The eject program allows the user to eject removable media (typically CD-ROMs, floppy disks or Iomega Jaz or Zip disks) using software control. **[Unnecessary]**

The bc package includes bc and dc. Bc is an arbitrary precision numeric processing arithmetic language. Dc is an interactive arbitrary precision stack based calculator, which can be used as a text mode calculator. **[Unnecessary]**

The mailcap file is used by the metamail program. Metamail reads the mailcap file to determine how it should display non-text or multimedia material. **[Unnecessary]**

The quota package contains system administration tools for monitoring and limiting users and or groups disk usage, per filesystem. **[Unnecessary]**

This is an Advanced Power Management daemon and utilities. It can watch your notebook's battery and warn all users when the battery is low. **[Unnecessary]**

Many laptop machines (and some non-laptops) support PCMCIA cards for expansion. Also known as "credit card adapters," PCMCIA cards are small cards for everything from SCSI support to modems. **[Unnecessary]**

Linuxconf is an extremely capable system configuration tool. Linuxconf provides four different interfaces for you to choose from command line, character-cell (like the installation program), an X Window System based GUI and a web-based interface. **[Unnecessary, buggy program]**

The getty_ps package contains the getty and uugetty programs, basic programs for accomplishing the login process on a Red Hat Linux system. Getty and uugetty are used to accept logins on the console or a terminal. **[Unnecessary]**

setconsole is a basic system utility for setting up the /etc/inittab, /dev/systty and /dev/console files to handle a new console. The console can be either the local terminal (i.e., directly attached to the system via a video card) or a serial console. **[Unnecessary]**

The isapnptools package contains utilities for configuring ISA Plug-and-Play (PnP) cards/boards, which are in compliance with the PnP ISA Specification Version 1.0a. **[Unnecessary]**

setserial is a basic system utility for displaying or setting serial port information. Setserial can reveal and allow you to alter the I/O port and IRQ that a particular serial device is using, and more. **[Unnecessary]**

The Sendmail program is a very widely used Mail Transport Agent (MTA). MTAs send mail from one machine to another. Sendmail is not a client program, which you use to read your email. Sendmail is a behind-the-scenes program, which actually moves your email over networks or the Internet to where you want it to go. **[Unnecessary, Security risks]**

The procmail program is used by Red Hat Linux for all local mail delivery. In addition to just delivering mail, procmail can be used for automatic filtering, presorting and other mail handling jobs. Procmail is also the basis for the SmartList mailing list processor. **[Unnecessary]**

The mailx package installs the /bin/mail program, which is used to send quick email messages (i.e., without opening up a full-featured mail user agent). Mail is often used in shell scripts. **[Unnecessary]**

VIII) Software that must be installed after installation of the Server

To be able to compile programs on your server you must install the following RPM's software. This part of the installation is very important and require that you install all related packages described bellow. Those software are on your Red Hat 6.1 Part 1 CD-ROM under RedHat/RPMS directory.

[root@deep]# **mount /dev/cdrom /mnt/cdrom/**
[root@deep]# **cd /mnt/cdrom/RedHat/RPMS/**

autoconf-2.13-5.noarch.rpm
m4-1.4-12.i386.rpm
automake-1.4-5.noarch.rpm
dev86-0.14.9-1.i386.rpm

bison-1.28-1.i386.rpm
byacc-1.9-11.i386.rpm
cdecl-2.5-9.i386.rpm
cpp-1.1.2-24.i386.rpm
cproto-4.6-2.i386.rpm
ctags-3.2-1.i386.rpm
cvs-1.10.6-2.i386.rpm
egcs-1.1.2-24.i386.rpm
ElectricFence-2.1-1.i386.rpm
flex-2.5.4a-7.i386.rpm
gdb-4.18-4.i386.rpm
gettext-0.10.35-13.i386.rpm
kernel-headers-2.2.12-20.i386.rpm
glibc-devel-2.1.2-11.i386.rpm
make-3.77-6.i386.rpm
patch-2.5-9.i386.rpm
pmake-2.1.33-5.i386.rpm
rcs-5.7-10.i386.rpm

Use command RPM like the following to install them. Command to install software is:

[root@deep]# **rpm -Uvh softwarenames-version.i386.rpm**

IX) Installed programs on your Server

This is the list of all installed programs that you must have in your computer after the complete installation of the Linux Server. This list must match exactly the **install.log** file located in your */tmp* directory or your could run under problem. Don't forget to install all programs listed above "Software that must be installed after installation of the Server" to be able to make compilation on your Server.

Installing setup.
Installing filesystem.
Installing basesystem.
Installing ldconfig.
Installing glibc.
Installing shadow-utils.
Installing mktemp.
Installing termcap.
Installing libtermcap.
Installing bash.
Installing MAKEDEV.
Installing SysVinit.
Installing XFree86-Mach64.
Installing chkconfig.
Installing apmd.
Installing arpwatch.
Installing ncurses.
Installing info.
Installing fileutils.
Installing grep.
Installing ash.
Installing at.
Installing authconfig.
Installing bc.
Installing bdflush.
Installing binutils.
Installing bzip2.

```
Installing sed.
Installing console-tools.
Installing e2fsprogs.
Installing rmt.
Installing cpio.
Installing cracklib.
Installing cracklib-dicts.
Installing crontabs.
Installing textutils.
Installing dev.
Installing diffutils.
Installing dump.
Installing ed.
Installing eject.
Installing etcskel.
Installing file.
Installing findutils.
Installing gawk.
Installing gd.
Installing gdbm.
Installing getty_ps.
Installing git.
Installing glib.
Installing gmp.
Installing gnupg.
Installing gpm.
Installing groff.
Installing gzip.
Installing hdparm.
Installing initscripts.
Installing isapnptools.
Installing kbdconfig.
Installing kernel.
Installing kernel-pcmcia-cs.
Installing kudzu.
Installing ld.so.
Installing less.
Installing libc.
Installing libpng.
Installing libstdc++.
Installing lilo.
Installing pwdb.
Installing pam.
Installing sh-utils.
Installing redhat-release.
Installing linuxconf.
Installing logrotate.
Installing losetup.
Installing lsof.
Installing mailcap.
Installing mailx.
Installing man.
Installing mingetty.
Installing mkbootdisk.
Installing mkinitrd.
Installing modutils.
Installing mount.
Installing mouseconfig.
Installing mt-st.
Installing ncompress.
Installing net-tools.
Installing netkit-base.
```

Installing newt.
Installing ntsysv.
Installing passwd.
Installing pciutils.
Installing perl.
Installing pidentd.
Installing procinfo.
Installing procmail.
Installing procps.
Installing psmisc.
Installing pump.
Installing python.
Installing quota.
Installing raidtools.
Installing readline.
Installing redhat-logos.
Installing rootfiles.
Installing rpm.
Installing sash.
Installing sendmail.
Installing setconsole.
Installing setserial.
Installing setuptool.
Installing shapecfg.
Installing slang.
Installing slocate.
Installing stat.
Installing sysklogd.
Installing tar.
Installing tcp_wrappers.
Installing tcpdump.
Installing tcsh.
Installing time.
Installing timeconfig.
Installing timed.
Installing tmpwatch.
Installing traceroute.
Installing utempter.
Installing util-linux.
Installing vim-common.
Installing vim-minimal.
Installing vixie-cron.
Installing which.
Installing zlib.

X) Put some colors on your terminal

Putting some colors on your terminal can help you to distinguish folders, files, archives, devices, symbolic links and executable file from others. My opinion is that colors help to make less errors and fast navigation on your system.

Edit the **profile** file (vi /etc/profile) and add the following lines:

```
# Enable Colour ls
eval `dircolors /etc/DIR_COLORS -b`
export LS_OPTIONS='-s -F -T 0 --color=yes'
```

Edit the **bashrc** file (vi /etc/bashrc) and add the line:

alias ls='ls --color=auto'

Then log in and out; after this, the new COLORS-environment variable is set, and your system will recognize that.

XI) Update of the lasted software's

Keep and update all software (especially network software) to the lasted versions, check the errata pages for the Red Hat Linux distribution, available at http://www.redhat.com/corp/support/errata/index.html. The errata pages are perhaps the best resource for fixing 90% of the common problems with Red Hat Linux. In addition, security holes for which a solution exists are generally on the errata page 24 hours after Red Hat has been notified. You should always check there first.

Software's that must be updated at this time for your Red Hat Linux 6.1 server are:

e2fsprogs-1.17-1.i386.rpm
pam-0.68-8.i386.rpm
Linux kernel 2.2.13 (The most important, always must be updated. See bellow for more information on building a custom kernel for your specific machine).

You can verify that software is installed on your system before make an update with the following command:

[root@deep]# **rpm -q software-name**

-Where software-name is the name of the software you wan to verify like XFree86 or telnet.

XII) For the maniacs

We know that Red Hat Linux by default install a lot thing that we're not necessary need. In our setup installation we 're removed the most important. Although that, some mark still exist on your system. For example, we're not installed the Xwindow system (XFree86) on our server or NIS program server but we can see that some folder already exist like /usr/X11R6/ or /var/nis. Those folders exist but are empty. It's safe to remove those folders and files since we are not installed those programs. Directories and files listed bellow can be removed safety from your system if you are followed our setup instruction above.

[root@deep]# rm -f /usr/bin/X11 ← symbolic links
[root@deep]# rm -f /etc/exports ← file for NFS server
[root@deep]# rm -f /etc/printcap ← file for the printer
[root@deep]# rm -rf /var/nis/ ← folder for NIS server
[root@deep]# rm -rf /etc/X11/ ← folder for Xwindow server
[root@deep]# rm -rf /usr/X11R6/ ← folder for Xwindow server
[root@deep]# rm -rf /etc/ppp/ ← folder for ppp modem connection
[root@deep]# rm -f /var/log/maillog ← log file for Mail server
[root@deep]# rm -f /var/log/spooler ← log file for printer
[root@deep]# rm -rf /var/spool/mail ← log file for Mail
[root@deep]# rm -rf /usr/doc/* ← documentation of different programs installed on your machine

Edit /etc/logrotate.d/syslog file and remove the lines related to **maillog** and **spooler**.
Edit /etc/rc.d/init.d/functions script file and remove **/usr/X11R6/bin** in the PATH environment line.

XIII) General system security

Linux Security

Overview

A UNIX system is only as secure as the administrator makes it. The more services you add, the more chances of introducing a security hole. Operating systems like SCO and others may actually be more prone to security breaches because they offer more services that are an integral part of how they operate, (in order to be more 'user friendly'). Linux itself is very stable and secure, but it in itself is distributed in many flavors. In one of the ongoing comparisons between RedHat and Slackware people have argued over which is more secure. When installing Linux, one should tend to install with the minimum, and then add only the ESSENTIAL items, reducing chances of an 'application' of having a security weakness. Linux is the most SECURE if properly implemented. If a weakness is apparent in the system, there are thousands of volunteers to point it out immediately, along with a fix. In a larger organization, such as some of the commercial products, they have a limited size of team members working on it. It is not always in their best interests to publicize any discoveries too loudly, and sometimes it takes a while before fixes trickle down the pipes into the releases or upgrades. Yes, they soon become available as patches, but most administrators of commercial products tend to use the tools available with the distribution only, with a false sense of comfort in that they have more professionally designed software. Mistakes can happen in programming at any level, but when you have 10's of thousand of people with the source code available to them, these mistakes are often discovered faster in an open source code environment. Of course, with 10's of thousands of people meddling with the source code, and what? 7 million copies of Linux out there now... there is a much better chance that someone will open a security hole too.

This document will discuss some of the general techniques used to secure your site. The following is for people attaching a computer to the Internet. Generally speaking services such as NFS / Samba, imap and pop will only need to be accessible to internal users, blocking external access greatly simplifies matters. The following is a list of features that can be used to help prevent attacks from external sources.

1. Basic Information

Let outsiders know as little as possible about your system. A simple finger to a victims system can reveal much about a system; How many users, when the admin is in, when do they work, who THEY are, who uses the system and personal information that might help an attacker guess a users password. You can use a powerful finger daemon and/or tcpd to limit who can connect to your system and how much they will know about your system.

Logs are the only way to know what's going on your Linux system. Of course, this assumes that an attacker has not corrupted your logs, but that aside. Full logging of all connections, can reveille so much about an attacker, and their attempts. If you have problems understanding the logs and what they are saying then get help to understand them. There is nothing wrong with someone helping you with something you don't understand. I get information every-now-and-then on something that I have made a mistake on and that I am told to clarify or fix something. To be arrogant is one thing to be stupid is another.

Limit the number of SUID root programs on your system. SUID root programs are programs when ran, run at the level of root (God in the UNIX world). Sometimes this is needed but many times not.

Since SUID root programs can do anything that root can they bear a high level of responsibility in following the golden rules of security programming. Sometimes they do, sometimes they don't and when they don't, users can sometimes get them to do things their not suppose to do. This is where exploits and come in. An exploit is a program or script that will get a SUID root program to do very bad stuff (Give root shells, grab password files, read other people's mail, delete files).

2. BIOS Security

Set a boot password.
Disallow booting from floppy drives and allow passwords to access some BIOS features.
Check your BIOS manual or look at it the next time you boot up.

3. Password

The starting point of our Linux security tour is the password. Many people keep their entire life on a computer and the only thing preventing others from seeing it is the eight-character string called a password. Not something one would call completely reliable. Contrary to popular belief, an uncrackable password does not exist. Given time and resources all passwords can be guessed either by social engineering or by brute force.

Since password cracking can be a time- and resource consuming art, make it hard for any cracker who has grabbed your password file. Running a password cracker on a weekly basis on your system is a good idea. This helps to find and replace passwords that are easily guessed or weak. Also, a password checking mechanism should be present to reject a weak password when first choosing a password or changing an old one. Character strings that are plain dictionary words, or are all in the same case, or do not contain numbers or special characters should not be accepted as a new password.

4. The /etc/exports file

If you are exporting filesystems using NFS, be sure to configure */etc/exports* file with the most restrictive access possible. This means not using wildcards, not allowing root write access, and mounting read-only wherever possible.

Edit the **exports** file (vi /etc/exports) and add:

For example:
/dir/to/export host1.mydomain.com(ro,root_squash)
/dir/to/export host2.mydomain.com(ro,root_squash)

-Where **/dir/to/export** is the directory you want to export, **host.mydomain.com** is the machine allowed to log in this directory, **ro** mean mounting read-only and **root_squash** for not allowing root write access in this directory.

For this change to take effect you will need to run **/usr/sbin/exportfs -a**

5. Disabling console program access

One of the simplest and commonest customizations is to entirely disable all console-equivalent access to programs like shutdown and halt. To do this, run:

[root@deep]# **rm -f /etc/security/console.apps/servicename**

-Where **servicename** is the name of the program to which you wish to disable console-equivalent access. Unless you use xdm, however, be careful not to remove the xserver file or no one but root will be able to start the X server. (If you always use xdm to start the X server, root is the only user that needs to start X, in which case you might actually want to remove the xserver file).

For example:
[root@deep]# **rm -f /etc/security/console.apps/halt**
[root@deep]# **rm -f /etc/security/console.apps/poweroff**
[root@deep]# **rm -f /etc/security/console.apps/reboot**
[root@deep]# **rm -f /etc/security/console.apps/shutdown**
[root@deep]# **rm -f /etc/security/console.apps/xserver** (if removed, root will be the only user able to start X).

6. Disabling all console access

In order to disable all console access, including program and file access, in the */etc/pam.d/* directory, comment out all lines that refer to **pam_console.so**. the following script will do the trick:

Create the **disabling.sh** script file (touch disabling.sh) and add the following lines inside:

```
cd /etc/pam.d
for i in * ; do
sed '/[^#].*pam_console.so/s/^/#/' < $i > foo && mv foo $i
done
```

Make this script executable with the following command and execute it:
[root@deep]# **chmod 700 disabling.sh**
[root@deep]# **./disabling.sh**

7. The /etc/inetd.conf file

Inetd, called the "super server", will load a network program based upon a request from the network. The inetd.conf file tell inetd which ports to listen to and what server to start for each port. As soon as you put your Linux system on ANY network the first thing to look at is what services you need to offer. Services that you do not need to offer should be disabled and mush better uninstalled so that you have one less thing to worry about and attackers have one less place to look for a hole. Look at your */etc/inetd.conf* file and see what services are being offered by your inetd. Disable any that you do not need by commenting them out (# at the beginning of the line), and then sending your inetd process a SIGHUP.

ENSURE that the permissions on this file are set to **600**.
[root@deep]# **chmod 600 /etc/inetd.conf**
ENSURE that the owner is **root**.
[root@deep]# **stat /etc/inetd.conf**

```
  File: "/etc/inetd.conf"
  Size: 2869        Filetype: Regular File
  Mode: (0600/-rw-------)        Uid: (    0/    root)  Gid: (    0/    root)
Device:  8,6   Inode: 18219    Links: 1
Access: Wed Sep  22   16:24:16    1999(00000.00:10:44)
Modify:  Mon Sep  20   10:22:44    1999(00002.06:12:16)
Change: Mon Sep  20   10:22:44    1999(00002.06:12:16)
```

Edit the **inetd.conf** file (vi /etc/inetd.conf) and disable services like:

ftp, telnet, shell, login, exec, talk, ntalk, imap, pop-2, pop-3, finger, auth, etc. unless you plan to use it. If it's turned off it's much less of a risk.

```
# To re-read this file after changes, just do a 'killall -HUP inetd'
#
#echo          stream    tcp       nowait    root        internal
#echo          dgram     udp       wait      root        internal
#discard       stream    tcp       nowait    root        internal
#discard       dgram     udp       wait      root        internal
#daytime       stream    tcp       nowait    root        internal
#daytime       dgram     udp       wait      root        internal
#chargen       stream    tcp       nowait    root        internal
#chargen       dgram     udp       wait      root        internal
#time          stream    tcp       nowait    root        internal
#time          dgram     udp       wait      root        internal
#
# These are standard services.
#
#ftp           stream    tcp       nowait    root        /usr/sbin/tcpd   in.ftpd -l -a
#telnet        stream    tcp       nowait    root        /usr/sbin/tcpd   in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
#shell         stream    tcp       nowait    root        /usr/sbin/tcpd   in.rshd
#login         stream    tcp       nowait    root        /usr/sbin/tcpd   in.rlogind
#exec          stream    tcp       nowait    root        /usr/sbin/tcpd   in.rexecd
#comsat        dgram     udp       wait      root        /usr/sbin/tcpd   in.comsat
#talk          dgram     udp       wait      root        /usr/sbin/tcpd   in.talkd
#ntalk         dgram     udp       wait      root        /usr/sbin/tcpd   in.ntalkd
#dtalk         stream    tcp       wait      nobody      /usr/sbin/tcpd   in.dtalkd
#
# Pop and imap mail services et al
#
#pop-2         stream    tcp       nowait    root        /usr/sbin/tcpd   ipop2d
#pop-3         stream    tcp       nowait    root        /usr/sbin/tcpd   ipop3d
#imap          stream    tcp       nowait    root        /usr/sbin/tcpd   imapd
#
# The Internet UUCP service.
#
#uucp          stream    tcp       nowait    uucp        /usr/sbin/tcpd  /usr/lib/uucp/uucico    -l
#
# Tftp service is provided primarily for booting.  Most sites
# run this only on machines acting as "boot servers." Do not uncomment
# this unless you *need* it.
#
#tftp          dgram     udp       wait      root        /usr/sbin/tcpd   in.tftpd
#bootps        dgram     udp       wait      root        /usr/sbin/tcpd   bootpd
#
# Finger, systat and netstat give out user information which may be
# valuable to potential "system crackers."  Many sites choose to disable
# some or all of these services to improve security.
#
#finger        stream    tcp       nowait    root        /usr/sbin/tcpd   in.fingerd
#cfinger       stream    tcp       nowait    root        /usr/sbin/tcpd   in.cfingerd
#systat        stream    tcp       nowait    guest       /usr/sbin/tcpd   /bin/ps -auwwx
#netstat       stream    tcp       nowait    guest       /usr/sbin/tcpd   /bin/netstat    -f inet
#
# Authentication
#
#auth          stream    tcp       nowait    nobody      /usr/sbin/in.identd    in.identd -l -e -o
#
# End of inetd.conf
```

NOTE: don't forget to send your inetd process a SIGHUP signal (killall –HUP inetd) after making change to your inetd.conf file.

[root@deep /root]# **killall -HUP inetd**

One more security measure you can take to secure the **inetd.conf** file is to set it immutable, using the **chattr** command. To set the file immutable simply:

[root@deep]# **chattr +i /etc/inetd.conf**

-And this will prevent any changes (accidental or otherwise) to the inetd.conf file. A file with the 'i' attribute cannot be modified: it cannot be deleted or renamed, no link can be created to this file and no data can be written to the file. Only the superuser can set or clear this attribute. If you wish to modify the inetd.conf file you will need to unset the immutable flag:

[root@deep]# **chattr -i /etc/inetd.conf**

8. TCP_WRAPPERS

By default Red Hat Linux allows all service requests. Using TCP_WRAPPERS makes securing your servers against outside intrusion is a lot simpler and painless then you would expect. Deny all hosts by putting "ALL: ALL@ALL, PARANOID" in */etc/hosts.deny* and explicitly list trusted hosts who are allowed to your machine in */etc/hosts.allow* file is the safest configuration. TCP_WRAPPERS is controlled from two files. The search stops at the first match.

/etc/hosts.allow
/etc/hosts.deny

- Access will be granted when a (daemon, client) pair matches an entry in the /etc/hosts.allow file.
- Otherwise, access will be denied when a (daemon, client) pair matches an entry in the /etc/hosts.deny file.
- Otherwise, access will be granted.

Edit the **hosts.deny** file (vi /etc/hosts.deny) and add the following line:

Access is denied by default.
 # Deny access to everyone.
ALL:ALL@ALL, PARANOID #Matches any host whose name does not match its address, see bellow.

-Which means all services, all locations, so any service not explicitly allowed is then blocked, unless they are permitted access by entries in the allow file.

NOTE: With the parameter **"PARANOID"**. If you are intended to run telnet or ftp services on your server, don't forget to add client machine name and IP address in your ***/etc/hosts*** file on the server or you can expect to wait several minutes for the DNS lookup to time out, before you get a login: prompt.

Edit the **hosts.allow** file (vi /etc/hosts.allow) and add the following line:

The explicitly authorized host are listed in the allow file.

For example:
sshd: 192.168.1.10/255.255.255.0 gate.openarch.com

-For your client machine: 192.168.1.10 are the IP address and gate.openarch.com the host name of one of your client allowed using sshd.

After your configuration is done, run the program **tcpdchk**.

[root@deep]# **tcpdchk**

-tcpdchk is the tcpd wrapper configuration checker. It examines your tcp wrapper configuration and reports all potential and real problems it can find.

9. The /etc/aliases file

The aliases file can easily be used to gain privileged status if it wrongly or carelessly administered. For example, many vendors used to ship systems with a "**decode"** alias in the aliases file. This practice is becoming less common.

The intention is to provide an easy way for users to transfer binary files using mail. At the sending site the user converts the binary to ASCII with "**uuencode"**, then mails the result to the "**decode"** alias at the receiving site. That alias pipes the mail message through the */usr/bin/uuencode* program, which converts the ASCII back into the original binary file.

Comment out the "**decode**" alias by placing a "**#**" at the beginning of the line. Similarly, every alias that executes a program -that you did not place there yourself and check completely- should be questioned and probably removed. For this change to take effect you will need to run:

[root@deep]# **/usr/bin/newaliases**

Edit the **aliases** file (vi /etc/aliases) and remove or comment out the following lines:

```
# Basic system aliases -- these MUST be present.
MAILER-DAEMON:  postmaster
postmaster:     root

# General redirections for pseudo accounts.
bin:            root
daemon:         root
#games:         root  ← remove or comment out.
#ingres:        root  ← remove or comment out.
nobody:         root
#system:        root  ← remove or comment out.
#toor:          root  ← remove or comment out.
#uucp:          root  ← remove or comment out.

# Well-known aliases.
#manager:       root  ← remove or comment out.
#dumper:        root  ← remove or comment out.
#operator:      root  ← remove or comment out.

# trap decode to catch security attacks
#decode:        root

# Person who should get root's mail
#root:          marc
```

Don't forget to run **/usr/bin/newaliases** for this change to take effect.

10. Prevent your system from responding to ping request

Preventing your system for responding to ping request can be a big improvement in your network security since no one can ping on your server and receive an answer.

An...
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
... should do the job too and your system won't respond to ping on any interface. You can add this line in your */etc/rc.d/rc.local* file so the command will be automatically set if your system reboot.

11. Don't let system issue file to be displayed

If you don't want your systems issue file to be displayed when people log in remotely, you can change the telnet option in your */etc/inetd.conf* file to look like:

telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd -h

-Adding the -h flag on the end will cause the daemon to not display any system information and just hit the user with a login: prompt. This hack is only necessary if you're using Telnet daemon.

12. The /etc/host.conf file

Edit the **host.conf** file (vi /etc/host.conf) and add the following lines:

Lookup names via DNS first then fall back to /etc/hosts.
order bind,hosts
We don't have machines with multiple IP addresses on the same card (like virtual server, IP Aliasing).
multi off
Check for IP address spoofing.
nospoof on

IP Spoofing: IP-Spoofing is a security exploit that works by tricking computers in a trust relationship that you are someone that you really aren't.

13. The /etc/securetty file

The */etc/securetty* file allows you to specify which **TTY** devices root is allowed to login on. The */etc/securetty* file is read by the login program (usually */bin/login*). Its format is a list of the **tty** devices names allowed, on all others root login is disallowed.

Disable any **tty** that you do not need by commenting them out (# at the beginning of the line).

Edit the **securetty** file (vi /etc/securetty) and comment out the following lines:

tty1
#tty2
#tty3
#tty4
#tty5
#tty6
#tty7
#tty8

-Which means root is only allowed to login on tty1.

14. Special accounts

DISABLE **ALL default vendor accounts** not necessary shipped with the Operating System. (This should be checked after each upgrade or installation). Linux provides these accounts for various system activities, which you may not need. If you do not need the accounts, remove them. The more accounts you have, the easier it is to access your system.

To delete user on your system, use the command:
[root@deep]# **userdel username**

To delete group on your system, use the command:
[root@deep]# **groupdel username**

Type the following commands on your terminal to delete users listed bellow:

[root@deep]# **userdel adm**
[root@deep]# **userdel lp**
[root@deep]# **userdel sync**
[root@deep]# **userdel shutdown**
[root@deep]# **userdel halt**
[root@deep]# **userdel mail** (delete this user if you don't use sendmail server, procmail and mailx).
[root@deep]# **userdel news**
[root@deep]# **userdel uucp**
[root@deep]# **userdel operator**
[root@deep]# **userdel games** (delete this user if you don't use X Window Server).
[root@deep]# **userdel gopher**
[root@deep]# **userdel ftp** (delete this user if you don't use ftp anonymous).

Type the following commands on your terminal to delete usersgroups listed bellow:

[root@deep]# **groupdel adm**
[root@deep]# **groupdel lp**
[root@deep]# **groupdel mail** (delete this group if you don't use sendmail server, procmail and mailx).
[root@deep]# **groupdel news**
[root@deep]# **groupdel uucp**
[root@deep]# **groupdel games** (delete this group if you don't use X Window Server).
[root@deep]# **groupdel dip**
[root@deep]# **groupdel pppusers**
[root@deep]# **groupdel popusers** (delete this group if you don't use pop server for email).
[root@deep]# **groupdel slipusers**

Add the necessary user to the system:

To add user on your system, use the command:
[root@deep]# **useradd username**

To add or change password for user on your system, use the command:
[root@deep]# **passwd username**

For example:
[root@deep]# **useradd admin**
[root@deep]# **passwd admin**

The output should look something like this.
Changing password for user admin
New UNIX password: somepasswd
passwd: all authentication tokens updated successfully

The immutable bit can be used to prevent accidentally deleting or overwriting a file that must be protected. It also prevents someone from creating a symbolic link to this file, which has been the source of attacks involving deleting */etc/passwd* or */etc/shadow*.

[root@deep]# **chattr +i /etc/passwd**
[root@deep]# **chattr +i /etc/shadow**
[root@deep]# **chattr +i /etc/group**
[root@deep]# **chattr +i /etc/gshadow**

15. Blocking anyone to su to root

If you don't want anyone to su to root or restrict su for certain users then add this to the top of your config su file in */etc/pam.d/* directory.

Edit the **su** file (vi /etc/pam.d/su) and add the following line in the file:

auth sufficient /lib/security/pam_rootok.so debug
auth required /lib/security/pam_wheel.so group=wheel

-Which mean only those who are a member of the wheel group can su to root. It also includes logging.

For example:
If you wan make user admin member of the wheel group and be able to su to root use the following command:

[root@deep]# **usermod -G10 admin**

-Which mean "G" is a list of supplementary groups, which the user is also a member of. "10" are the numerical value of the user's ID "wheel". "admin" is the user we wan to add to wheel group.

16. Resource limits

Set resource limits on all your users so they can't perform denial of service attacks (number of processes, amount of memory, etc). These limits will have to be setup for the user when he or she logs in. For example, limits for all users on your system might look like this.

Edit the **limits.conf** file (vi /etc/security/limits.conf) and add:

```
*       hard    core    0
*       hard    rss     5000
*       hard    nproc   20
```

You must also edit */etc/pam.d//login* file and add or verify the existence of this line:

session required /lib/security/pam_limits.so

-This says to prohibit the creation of core files "core 0", restrict the number of processes to 50 "nproc 50", and restrict memory usage to 5M "rss 5000" for everyone. All of the above only concern users who have entered through the login prompt.

17. More control on mounting a file system

You can have more control on mounting a file system like */home* and */tmp* with some nifty options like noexec, nodev, and nosuid. This can be setup in */etc/fstab*:

Edit the **fstab** file (vi /etc/fstab) and add depending of your needs:

```
/dev/sda11      /tmp    ext2    nosuid,nodev,noexec 1 2
/dev/sda6       /home   ext2    nosuid,nodev 1 2
```

-Which mean for <*nodev*> do not interpret character or block special devices on the file system, for <*nosuid*> do not allow set-user-identifier or set-group-identifier bits to take effect and for <*noexec*> do not allow execution of any binaries on the mounted file system.

18. Prevent you from running rm –rf * command on directories

Doing a: **touch -- -i** in a directory that's important to you will prevent you from running rm -rf * in that directory non-interactively.

For example:
[root@deep]# **cd /usr/**
[root@deep]# **touch -- -i**

19. Shell logging

For the root account I would highly recommend setting the HISTFILESIZE and HISTSIZE in */etc/profile* file to a low value such as **20**. Also set the file "**.bash_history"** in all user directory non world readable and set the configurations files in the user home directory to immutable using the **chattr** command to append only.

Edit the **profile** file (vi /etc/profile) and change the line to:

HISTFILESIZE=**20**
HISTSIZE=**20**

cd `/ (to switch to home directory)

[root@deep]# **chmod 640 .bash_history**
[root@deep]# **chattr +A .bash_history**
[root@deep]# **chattr +A .bash_logout**
[root@deep]# **chattr +A .bash_profile**
[root@deep]# **chattr +A .bashrc**

20. File Permission corrections

sda10 is our swap partition on the system for this example, if you are an IDE disk it will be **hdaN**, if you are a SCSI disk it will be **sdaN** like the example bellow. To see which device number represent your swap, edit the */etc/fstab* file and check inside.

/dev/sda10/ ensures that the permissions are set to **600**
[root@deep]# **stat /dev/sda10**
[root@deep]# **chmod 600 /dev/sda10**

Make this change permanent by adding this line to your */etc/rc.d/rc.local* file.
chmod 600 /dev/sda10

21. The /etc/lilo.conf file

a) Add: **restricted**
Requires a password to be used if boot time options (such as "linux single") are passed to the boot loader. Make sure you use this one on each image.

b) Add: **password=some_password**
Requires user to input a password, used in conjunction with restricted, also make sure lilo.conf is no longer world readable, or any user will be able to read the password. Here is an example of lilo.conf file.

Edit the **lilo.conf** file (vi /etc/lilo.conf) and add:

boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=00 ← change this line to 00.
Default=linux
restricted ← add this line.
password=some_password ← add this line.
image=/boot/vmlinuz-2.2.12-20
label=linux
initrd=/boot/initrd-2.2.12-10.img
root=/dev/sda6
read-only

[root@deep]# **chmod 600 /etc/lilo.conf** (will be no longer world readable).
[root@deep]# **/sbin/lilo -v** (to update the lilo.conf file).

One more security measure you can take to secure the lilo.conf file is to set it immutable, using the **chattr** command. To set the file immutable simply:

[root@deep]# **chattr +i /etc/lilo.conf** and this will prevent any changes (accidental or otherwise) to the lilo.conf file. If you wish to modify the lilo.conf file you will need to unset the immutable flag: **chattr -i /etc/lilo.conf**

c) Add: **timeout=X**
This controls how long (in tenths of seconds) LILO waits for user input before booting to the default selection. One of the requirements of C2 security is that this interval be set to 0 (obviously a dual boot machines blows most security out of the water). It is a good idea to set this to 0 unless the system dual boots something else.

22. Disable the Control-Alt-Delete keyboard shutdown command

This is pretty important if you don't have the best physical security on the box: To do this, edit the */etc/inittab* file and change the line:

[root@deep]# vi /etc/inittab

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

Now, for the system to understand the change type in the following at a prompt:

[root@deep]# **/sbin/init q**

23. Fix the permissions on the /etc/rc.d/init.d script files

[root@deep]# **chmod -R 700 /etc/rc.d/init.d/***

-Which means just root is allowed to Read, Write, and Execute scripts files on this directory.

24. The /etc/rc.d/rc.local file

By default, when you login to a Linux box, it tells you the Linux distribution name, version, kernel version, and the name of the server. This is giving away too much info. We rather just prompt users with a "Login:" prompt.

To do this, Edit the *"/etc/rc.d/rc.local"* file and Place "#" in front of the following lines like shown

```
--
# This will overwrite /etc/issue at every boot.  So, make any changes you
# want to make to /etc/issue here or you will lose them when you reboot.
#echo "" > /etc/issue
#echo "$R" >> /etc/issue
#echo "Kernel $(uname -r) on $a $(uname -m)" >> /etc/issue
#
#cp -f /etc/issue /etc/issue.net
#echo >> /etc/issue
--
```

Then, do the following:
[root@deep]# **rm -f /etc/issue**
[root@deep]# **rm -f /etc/issue.net**
[root@deep]# **touch /etc/issue**
[root@deep]# **touch /etc/issue.net**

25. Bits from root-owned programs

Remove the 's' bits from root-owned programs that won't require such privilege. This can be accomplished by executing the command **'chmod a-s'** with the name(s) of the offending files as it's arguments.

Such programs include, but aren't limited to:

1.programs you never use.
2.programs that you don't want any non-root users to run.
3.programs you use occasionally, and don't mind having to su (1) to root to run.

We've placed an asterisk (*) next to each program we personally might disable. Remember that your system needs some suid root programs to work properly, so be careful.

[root@deep]# **find / -type f \(-perm -04000 -o -perm -02000 \) \-exec ls –lg {} \;**

```
-rwsr-xr-x    1 root     root        33120 Mar 21  1999 /usr/bin/at
*-rwsr-xr-x   1 root     root        30560 Apr 15 20:03 /usr/bin/chage
*-rwsr-xr-x   1 root     root        29492 Apr 15 20:03 /usr/bin/gpasswd
-rwsr-xr-x    1 root     root         3208 Mar 22  1999 /usr/bin/disable-paste
-rwxr-sr-x    1 root     man         32320 Apr  9  1999 /usr/bin/man
```

```
-r-s--x--x    1 root     root        10704  Apr 14   17:21  /usr/bin/passwd
-rws--x--x    2 root     root       517916  Apr  6    1999  /usr/bin/suidperl
-rws--x--x    2 root     root       517916  Apr  6    1999  /usr/bin/sperl5.00503
-rwxr-sr-x    1 root     mail        11432  Apr  6    1999  /usr/bin/lockfile
-rwsr-sr-x    1 root     mail        64468  Apr  6    1999  /usr/bin/procmail
-rwsr-xr-x    1 root     root        21848  Aug 27   11:06  /usr/bin/crontab
-rwxr-sr-x    1 root     slocate     15032  Apr 19   14:55  /usr/bin/slocate
*-r-xr-sr-x   1 root     tty          6212  Apr 17   11:29  /usr/bin/wall
*-rws--x--x   1 root     root        14088  Apr 17   12:57  /usr/bin/chfn
*-rws--x--x   1 root     root        13800  Apr 17   12:57  /usr/bin/chsh
*-rws--x--x   1 root     root         5576  Apr 17   12:57  /usr/bin/newgrp
*-rwxr-sr-x   1 root     tty          8392  Apr 17   12:57  /usr/bin/write
-rwsr-x---    1 root     squid       14076  Oct  7    14:48  /usr/lib/squid/pinger
-rwxr-sr-x    1 root     utmp        15587  Jun  9    09:30  /usr/sbin/utempter
*-rwsr-xr-x   1 root     root         5736  Apr 19   15:39  /usr/sbin/usernetctl
*-rwsr-xr-x   1 root     bin         16488  Jul  6   09:35  /usr/sbin/traceroute
-rwsr-sr-x    1 root     root       299364  Apr 19   16:38  /usr/sbin/sendmail
-rwsr-xr-x    1 root     root        34131  Apr 16   18:49  /usr/libexec/pt_chown
-rwsr-xr-x    1 root     root        13208  Apr 13   14:58  /bin/su
*-rwsr-xr-x   1 root     root        52788  Apr 17   15:16  /bin/mount
*-rwsr-xr-x   1 root     root        26508  Apr 17   20:26  /bin/umount
*-rwsr-xr-x   1 root     root        17652  Jul  6   09:33  /bin/ping
-rwsr-xr-x    1 root     root        20164  Apr 17   12:57  /bin/login
*-rwxr-sr-x   1 root     root         3860  Apr 19   15:39  /sbin/netreport
-r-sr-xr-x    1 root     root        46472  Apr 17   16:26  /sbin/pwdb_chkpwd
```

```
[root@deep]# chmod a-s /usr/bin/chage
[root@deep]# chmod a-s /usr/bin/gpasswd
[root@deep]# chmod a-s /usr/bin/wall
[root@deep]# chmod a-s /usr/bin/chfn
[root@deep]# chmod a-s /usr/bin/chsh
[root@deep]# chmod a-s /usr/bin/newgrp
[root@deep]# chmod a-s /usr/bin/write
[root@deep]# chmod a-s /usr/sbin/usernetctl
[root@deep]# chmod a-s /usr/sbin/traceroute
[root@deep]# chmod a-s /bin/mount
[root@deep]# chmod a-s /bin/umount
[root@deep]# chmod a-s /bin/ping
[root@deep]# chmod a-s /sbin/netreport
```

If you wan to know what those programs do, make a man program-name and read.

26. Find all files with the SUID/SGID bit enabled

SUID and SGID files on your system are a potential security risk, and should be monitored closely. Because these programs grant special privileges to the user who is executing them, it is necessary to ensure that insecure programs are not installed. A favorite trick of crackers is to exploit SUID "root" programs, then leave a SUID program as a backdoor to get in the next time, even if the original hole is plugged. Find all SUID and SGID programs on your system, and keep track of what they are, so you are aware of any changes, which could indicate a potential intruder. Use the following command to find all SUID/SGID programs on your system.

```
[root@deep]# find / -type f \( -perm -04000 -o -perm -02000 \) \-exec ls -lg {} \; > suid-sgid-results
```

27. Find group and World Writable files and directories

Particularly system files, can be a security hole if a cracker gain access to your system and modifies them. Additionally, world-writable directories are dangerous, since they allow a cracker

to add or delete files as he wishes. In the normal course of operation, several files will be writable, including some from */dev*, and symbolic links. To locate all world-writable files on your system, use the following command:

[root@deep]# **find / -type f \(-perm -2 -o -perm -20 \) -exec ls -lg {} \; > ww-files-results**
[root@deep]# **find / -type d \(-perm -2 -o -perm -20 \) -exec ls -ldg {} \; > ww-directories-results**

28. Unowned files

Unowned files may also be an indication an intruder has accessed your system. Don't permit any unowned file. You can locate files on your system that do not have an owner, or belong to a group with the command:

[root@deep]# **find / -nouser -o -nogroup > unowed-results**

NOTE: Once again, files reported under the */dev* directory don't count.

29. Finding .rhosts files

The ".rhosts" files should be a part of your regular system administration duties, as these files should not be permitted on your system. Remember that a cracker only needs one insecure account to potentially gain access to your entire network. You can locate all "**.rhosts"** files on your system with the following command:

[root@deep]# **find /home -name .rhosts > rhost-results**

30. System has been compromised

If you believe that your system has been compromised, contact CERT ® Coordination Center or your representative in FIRST (Forum of Incident Response and Security Teams).

Internet Email: cert@cert.org
CERT Hotline: (+1) 412-268-7090
Facsimile: (+1) 412-268-6989
CERT/CC personnel answer 8:00 a.m. – 8:00 p.m. EST (GMT –5)/EDT (GMT –4)) on working days; they are on call for emergencies during other hours and on weekends and holidays.

XIV) General system optimization

Linux Optimization

1. The /etc/profile file

The */etc/profile* file contains system wide environment stuff and startup programs. To squeeze the most performance from your x86 programs, you can use full optimization when compiling with the **-O6** flag. Many programs contain -O2 in the Makefile. -O6 is the highest level of optimization. It will increase the size of what it produces, but it runs faster. When compiling, use **-fomit-frame-pointer**. This will use the stack for accessing variables. Unfortunately debugging is almost impossible with this option. If you happen to have a PentiumPro or other high end CPU, you could

use the -mcpu=cpu_type and -march=cpu_type switch. This will optimize for the CPU listed to the best of GCC's ability. However, the resulting code will only be runnable on the indicated CPU or higher.

In the ***/etc/profile*** file, put this line for a Pentium II and III Pro Processor family
CFLAGS='-O6 -malign-double -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions'

In the ***/etc/profile*** file, put this line for a Pentium Processor family
CFLAGS='-O6 -malign-double -mcpu=pentium -march=pentium -fomit-frame-pointer -fno-exceptions'

- and a bit further down, add " **CFLAGS LANG LESSCHARSET**" to the "export"-line:
export PATH PS1 HOSTNAME HISTSIZE HISTFILESIZE USER LOGNAME MAIL INPUTRC
CFLAGS LANG LESSCHARSET

Then log in and out; after this, the new CFLAGS-environment variable is set, and Samba's and other "configure"-tool will recognize that. Pentium (Pro/II/III) optimizations will only work with egcs or pgcc compilers. egcs compiler is already installed on your Server by default so you don't need to think about.

-malign-double
Control whether GCC aligns double, long double, and long long variables on a two-word boundary or a one-word boundary. Aligning double variables on a two-word boundary will produce code that runs somewhat faster on a `Pentium' at the expense of more memory.

-mcpu=cpu_type
Assume the defaults for the machine type cpu type when scheduling instructions. While picking a specific cpu type will schedule things appropriately for that particular chip, the compiler will not generate any code that does not run on the i386 without the `-march=cpu_type' option being used. `i586' is equivalent to `pentium' and `i686' is equivalent to `pentiumpro'. `k6' is the AMD chip as opposed to the Intel ones.

-march=cpu_type
Generate instructions for the machine type cpu type. The choices for cpu type are the same as for `-mcpu'. Moreover, specifying `-march=cpu_type' implies `-mcpu=cpu_type'.

-fomit-frame-pointer
Don't keep the frame pointer in a register for functions that don't need one. This avoids the instructions to save, set up and restore frame pointers; it also makes an extra register available in many functions. It also makes debugging impossible on most machines.

2. The bdflush parameter

This documentation is for the sysctl files in */proc/sys/vm* and is valid for Linux kernel version 2.2. The files in this directory can be used to tune the operation of the virtual memory (VM) subsystem of the Linux kernel, and one of the files (bdflush) also has a little influence on disk usage. This file (bdflush) controls the operation of the bdflush kernel daemon. We generally use this command to improve filesystem performance.

echo "100 1200 128 512 15 5000 500 1884 2">/proc/sys/vm/bdflush

Be changing some values from the default and the system seems more responsive, ie: it waits a little more to write to disk and thus avoids some disk access contention.

Add the above commands to ***/etc/rc.d/rc.local*** file and you'll not have to type it again the next time if you reboot your system.

Look at */usr/src/linux/Documentation/sysctl/vm.txt* for more information on how to improve kernel parameters related to virtual memory, disk cache, swap, etc.

3. The ip_local_port_range parameter

This documentation is for the sysctl files in */proc/sys/net/ipv4/ip_local_port_range* and is valid for Linux kernel version 2.2. ip_local_port_range - 2 INTEGERS.

Defines the local port range that is used by TCP and UDP to choose the local port. The first number is the first (port), the second the last local port number. For high-usage systems change this to 32768-61000.

echo "32768 61000" > /proc/sys/net/ipv4/ip_local_port_range

Add the above commands to ***/etc/rc.d/rc.local*** file and you'll not have to type it again the next time if you reboot your system.

4. The /etc/nsswitch.conf file

[root@deep]# vi /etc/nsswitch.conf

Change the "hosts" line to read

"hosts: dns files"

Also.. I would recommend to DELETE all instances of NIS from each line of this file UNLESS you *ARE* using NIS! The result must look like this:

passwd: **files**
shadow: **files**
group: **files**
hosts: dns files
services: **files**
networks: **files**
protocols: **files**
rpc: **files**
ethers: **files**
netmasks: **files**
bootparams: **files**
automount: **files**
aliases: **files**

5. The /proc filesystem

This documentation is for the sysctl files in */proc/sys/fs* and is valid for Linux kernel version 2.2. The files in this directory can be used to tune and monitor miscellaneous and general things in the operation of the Linux kernel. Since some of the files _can_ be used to screw up your system, it is advisable to read both documentation and source before actually making adjustments.

Increase the value of ***/proc/sys/fs/file-max*** to something reasonable like 256 for every 4M of RAM you have: i.e. for a 128 M machine, set it to 8192 (128/4=32 32*256=8192). As above, also

increase the ***/proc/sys/fs/inode-max*** to a value roughly 3 to 4 times (8192*4=32768) the number of open files. This is because the number of inodes open is at least one per open file, and often much larger than that for large files.

The canonical command to change anything in the */proc* hierarchy is (as root**) echo "newvalue" >/proc/file/that/you/want/to/change**, so for this item the command line is

echo "8192" >/proc/sys/fs/file-max
echo "32768" >/proc/sys/fs/inode-max

The above technique or modifying the constants in the kernel sources. Not usually the right answer because that will not survive a new kernel source tree. One of the best techniques is to add the above commands to ***/etc/rc.d/rc.local*** file.

[root@deep]# vi /etc/rc.d/rc.local and add at the end of this file the following lines

echo "8192" >/proc/sys/fs/file-max (If you have a 128MB of RAM in your system).
echo "32768" >/proc/sys/fs/inode-max (If you have a 128MB of RAM in your system).

The exact number will vary from the above formula based on what you are actually doing with the machine. A file server or web server need a lot of open files, for instance, but a compute server does not.

Very large memory systems, especially 512 Megabytes or larger, probably should not have more than 50,000 open files and 150,000 open inodes.

The value in **file-max** denotes the maximum number of file handles that the Linux kernel will allocate. When you get a lot of error messages about running out of file handles, you might want to raise this limit. The default value is 4096.

The value in **inode-max** denotes the maximum number of inode handlers. This value should be 3 to 4 times larger than the value in file-max, since stdin, stdout, and network sockets also need an inode struct to handle them. If you regularly run out of inodes, you should increase this value.

6. The ulimit parameter

NOTE: Linux itself has a "Max Processes" per user limit. Add this to your root "**.bashrc"** file or whatever script your particular shell uses (in our case we'll use the /etc/profile file to make it on in all our system, process and users):

Edit the **profile** file (vi /etc/profile) and add the following line:
ulimit -u unlimited

You must exit and re-login. To verify that you are ready to go, make sure that when you type **ulimit -a** as root, it shows "unlimited" next to **max user processes**

[root@deep]# **ulimit -a**

core file size (blocks)	1000000
data seg size (kbytes)	unlimited
file size (blocks)	unlimited
max memory size (kbytes)	unlimited
stack size (kbytes)	8192
cpu time (seconds)	unlimited
max user processes	**unlimited ← this line.**
pipe size (512 bytes)	8

open files 1024
virtual memory (kbytes) 2105343

NOTE: you may also do *ulimit -u unlimited* at the command prompt instead of adding it to the */etc/profile* file, but I always forgot, so I just added it in the */etc/profile* file as a safety net.

7. Increases the system limit on open files

Increases the current process' limit. I verified that a process on Red Hat 6.0 (2.2.5) could open at least 31000 file descriptors this way. Another fellow has verified that a process on 2.2.12 can open at least 90000 file descriptors this way (with appropriate limits). The upper bound seems to be available memory.

Edit the **profile** file (vi /etc/profile) and add the following line:
ulimit -n 90000

You must exit and re-login. To verify that you are ready to go, make sure that when you type **ulimit -a** as root, it shows "90000" next to **open files**.

[root@deep]# **ulimit -a**

core file size (blocks) 1000000
data seg size (kbytes) unlimited
file size (blocks) unlimited
max memory size (kbytes) unlimited
stack size (kbytes) 8192
cpu time (seconds) unlimited
max user processes unlimited
pipe size (512 bytes) 8
open files **90000 ← this line.**
virtual memory (kbytes) 2105343

NOTE: In older 2.2 kernels, though, the number of open files per process is still limited to 1024, even with the above changes.

8. The atime attribute

In addition to the information about when files were created and last modified, Linux also records when a file was last accessed. This information is not particularly useful, and there is a cost associated with recording it. The ext2 filesystem allows the superuser to mark individual files such that their last access time is not recorded. This may lead to significant performance improvements when running find, and may also be useful on often accessed frequently changing files such as the contents of */var/spool/news*.

To set the attribute, use:

[root@deep]# **chattr +A filename**

For a whole directory tree, does something like:

[root@deep /root]# **chattr -R +A /var/spool/**
[root@deep /root]# **chattr -R +A /cache/**
[root@deep /root]# **chattr -R +A /home/httpd/ona/**

9. The noatime attribute

Linux has a mount option for filesystems call **noatime**. This option can be added to the mount options field in */etc/fstab*. When a filesystem is mounted with this option, read accesses to the file will no longer result in an update to the atime information associated with the file. The atime info is generally not all that useful, so the lacks of updates to this field are not often relevant. The importance of the **noatime** setting is that it eliminates the need to make writes to the filesystem for files, which are simply being read. Since writes tend to be somewhat expensive, this can result in measurable performance gains. Note that the wtime information will continue to be updated anytime the file is written to.

[root@deep]# vi /etc/fstab and add for example in the line:

E.I: /dev/sda7 /chroot ext2 defaults,**noatime** 1 2

Reboot your system and then test your results with the command:
[root@deep]# **cat /proc/mounts**

10. TCP/IP Stack specific

Red Hat Linux, out of the box, does NOT optimize the TCP/IP window size. This can make a BIG difference with performance. For more information, check out: RFC 1106 - High Latency WAN links - Section 4.1 and RFC 793 - Transmission Control Protocol.

Edit "*/etc/sysconfig/network-scripts/ifup*" and around lines 110, 112, 117, 125, and 134, find the lines

```
110:   "route add -net ${NETWORK} netmask ${NETMASK} ${DEVICE}"
112:   "route add -host ${IPADDR} ${DEVICE}"
117:   "route add default gw ${GATEWAY} metric 1 ${DEVICE}"
125:   "route add default gw ${GATEWAY} ${DEVICE}"
134:   "route add default gw $gw ${DEVICE}"
```

And change them to

```
110:   "route add -net ${NETWORK} netmask ${NETMASK} window 8192 ${DEVICE}"
112:   "route add -host ${IPADDR} window 8192 ${DEVICE}"
117:   "route add default gw ${GATEWAY} window 8192 metric 1 ${DEVICE}"
125:   "route add default gw ${GATEWAY} window 8192 ${DEVICE}"
134:   "route add default gw $gw window 8192 ${DEVICE}"
```

11. The swap partition

Try to put your swap partitions near the beginning of your drive. The beginning of the drive is physically located on the outer portion of the cylinder, so the read/write head can cover much more ground per revolution. I typically see partitions placed at the end work 3MB/s slower using hdparm -t.

12. Tuning IDE Hard Disk Performance

2x performance increases have been reported on massive disk I/O operations (like cloning disks) by setting the IDE drivers to use DMA and 32-bit transfers. The kernel seems to use more conservative settings unless told otherwise.

The commands are
[root@deep]# **/sbin/hdparm -c 1 /dev/hda** (or hdb, hdc etc).

To use 32-bit I/O over the PCI buses. (The hdparm (8) manpage says that you may need to use -c 3 for some chipsets).

Use:
[root@deep]# **/sbin/hdparm -d 1 /dev/hda** (or hdb, hdc etc).

To enable DMA. This may depend on support for your motherboard chipset being compiled into your kernel.

You can test the results of your changes by running hdparm in performance test mode:
[root@deep]# **/sbin/hdparm -t /dev/hda** (or hdb, hdc etc).

When you've found the optimal settings, you should consider doing a
[root@deep]# **/sbin/hdparm -k 1 /dev/hda** (or hdb, hdc etc).

To keep these settings across an IDE reset. I've seen the kernel reset the IDE controller occasionally and if you don't set -k 1, the other settings will be reset to defaults and you'll lose all your performance gains.

XV) Recompiling the Kernel

The first thing to do next is build a kernel that best suits your system. It's very simple to do but, in any case, refer to the README file in */usr/src/linux/* or the Kernel HOWTO.

Linux Kernel

Overview:

When configuring your kernel only compile in code you need and use. Four main reason come into mind; the Kernel will be faster (Less code to run), you will have more memory (Kernel parts are NEVER swapped to the virtual memory), more stable (Ever probed for a non-existent card?), unnecessary parts can be used by an attacker gain access to the machine or other machines.

These installation instructions assume

Commands are Unix-compatible.
The source path is /usr/src.
Installations were tested on RedHat Linux 6.1 Server.
All steps in the installation will happen in superuser account "root".
Lasted Kernel version number is 2.2.13

Packages

Kernel Homepage: http://www.kernelnotes.org/
You must be sure to download: linux-2.2.13.tar.gz

You can configure the Linux kernel in one of three ways. The first method is to use the **make config** command. It provides you with a text-based interface for answering all the configuration options. You are prompted for all the options you need to set up your kernel. The second method is to use the **make menuconfig** command, which provides all the kernel options in an easy-to-

use menu. The third is to use the **make xconfig** command, which provides a full graphical interface to all the kernel options. For configuration in this chapter, you will use the *make config* command because we are not installed the Xfree86 window Interface on our server.

Making an emergency boot floppy

The first pre-install step is to make an emergency boot floppy (if you haven't made one already). You can simply do this with the mkbootdisk command.

First find out what kernel, you are currently using. Check out your */etc/lilo.conf* file and see which image was booted from. On my example, I have the following in the file.

```
[root@deep]# cat /etc/lilo.conf
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.12-20
        label=linux
        root=/dev/sda6
        initrd=/boot/initrd-2.2.12-20.img
        read-only
```

Now you will need to find the image that you booted from. On a standard install, it will be the one-labeled linux. The above example shows that the machine booted using the */boot/vmlinuz-2.2.12-20* kernel. Now simply put a formatted 1.44 floppy in your system, and make sure you have logged in as root.

[root@deep]# **mkbootdisk --device /dev/fd0 2.2.12-20**
Insert a disk in /dev/fd0. Any information on the disk will be lost.
Press <Enter> to continue or ^C to abort:

Following these guidelines, you will now have a boot floppy with a known working kernel in case of problems with the upgrade. I recommend rebooting the system with the floppy to make sure that the floppy works correctly.

Optimization

Decompress the tarball (tar.gz).

[root@deep]# cp linux-version.tar.gz /usr/src/
Cd into /usr/src/
[root@deep]# rm -rf linux (or uninstall kernel-headers-version.i386.rpm, kernel-version.i386.rpm if it's a fresh install).
If kernel RPM so make [root@deep]# **rpm -e --nodeps kernel-headers kernel**
[root@deep]# **rm -rf /usr/src/linux-2.2.12/**
[root@deep]# **rm -rf /lib/modules/2.2.12-20/**
[root@deep]# **tar xzpf linux.version.tar.gz**
[root@deep]# **chown -R 0.0 /usr/src/linux/**
[root@deep]# **rm -f linux-version.tar.gz**

Increase the Tasks

To increase the number of tasks allowed, you may need to edit the */usr/src/linux/include/linux/tasks.h* file and change the following parameters.

[root@deep]# vi /usr/src/linux/include/linux/tasks.h
Change NR_TASKS from 512 to **3072**
Change MIN_TASKS_LEFT_FOR_ROOT from 4 to **24**

Edit the **Makefile** file (vi /usr/src/linux/Makefile) and change the line:
CFLAGS = -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer
To:
CFLAGS = -Wall -Wstrict-prototypes -O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions

Edit the **Makefile** file (vi /usr/src/linux/Makefile) and change the line:
HOSTCFLAGS = -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer
To:
HOSTCFLAGS = -Wall -Wstrict-prototypes -O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions

-Which turns on a few optimization tricks that may or may not work with all kernels.

Compilation

Make sure your */usr/include/asm*, */usr/include/linux*, and */usr/include/scsi* directories are just symlinks to the kernel sources: Type the following commands on your terminal.

[root@deep]# **cd /usr/include**
[root@deep]# **rm -rf asm linux scsi**
[root@deep]# **ln -s /usr/src/linux/include/asm-i386 asm**
[root@deep]# **ln -s /usr/src/linux/include/linux linux**
[root@deep]# **ln -s /usr/src/linux/include/scsi scsi**

Make sure you have no stale .o files and dependencies lying around: Type the following commands on your terminal.

[root@deep]# **cd /usr/src/linux/**
[root@deep]# **make mrproper**

You should now have the sources correctly installed.

[root@deep]# **cd /usr/src/linux/** (if you are not already in this directory).
[root@deep]# **make config**

Code maturity level options:
Prompt for development and/or incomplete code/drivers (CONFIG_EXPERIMENTAL) [N]

Processor type and features:
Processor family (CONFIG_M386) [Ppro/6x86MX]
Maximum physical Memory (CONFIG_1GB) [1GB]
Math emulation (CONFIG_MATH_EMULATION) [N]
MTRR Memory Type Range Register support (CONFIG_MTRR) [N]
Symmetric multi-processing support (CONFIG_SMP) [Y] **N**

Loadable module support:
Enable loadable module support (CONFIG_MODULES) [Y] **N**

General setup:
Networking support (CONFIG_NET) [Y]
PCI support (CONFIG_PCI) [Y]
PCI quirks (CONFIG_PCI_QUIRKS) [Y] **N**
Backward-compatible /proc/pci (CONFIG_PCI_OLD_PROC) [Y] **N**
MCA support (CONFIG_MCA) [N]
SGI Visual Workstation support (CONFIG_VISWS) [N]
System V IPC (CONFIG_SYSVIPC) [Y]

BSD Process Accounting (CONFIG_BSD_PROCESS_ACCT) [N]
Sysctl support (CONFIG_SYSCTL) [Y]
Kernel support for a.out binaries (CONFIG_BINFMT_AOUT) [Y]
Kernel support for ELF binaries (CONFIG_BINFMT_ELF) [Y]
Kernel support for MISC binaries (CONFIG_BINFMT_MISC) [Y]
Parallel port support (CONFIG_PARPORT) [N]
Advanced Power Management BIOS supports (CONFIG_APM) [N]

Plug and Play support:
Plug and Play support (CONFIG_PNP) [N]

Block devices:
Normal PC floppy disk support (CONFIG_BLK_DEV_FD) [Y]
Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support (CONFIG_BLK_DEV_IDE) [Y]
Use old disk-only driver on primary interface (CONFIG_BLK_DEV_HD_IDE) [N]
Include IDE/ATA-2 disk support (CONFIG_BLK_DEV_IDEDISK) [Y]
Include IDE/ATAPI CDROM support (CONFIG_BLK_DEV_IDECD) [Y]
Include IDE/TAPE support (CONFIG_BLK_DEV_IDETAPE) [N]
Include IDE/FLOPPY support (CONFIG_BLK_DEV_IDEFLOPPY) [N]
SCSI emulation support (CONFIG_BLK_DEV_IDESCSI) [N]
CMD640 chipset bugfix/support (CONFIG_BLK_DEV_CMD640) [Y] **N**
RZ1000 chipset bugfix/support (CONFIG_BLK_DEV_RZ10000 [Y] **N**
Generic PCI IDE chipset support (CONFIG_BLK_DEV_IDEPCI) [Y]
Generic PCI bus-master DMA support (CONFIG_BLK_DEV_IDEDMA) [Y]
Boot off-board chipsets first support (CONFIG_BLK_DEV_OFFBOARD) [N]
Use DMA by default when available (CONFIG_IDEDMA_AUTO) [Y]
Other IDE chipset support (CONFIG_IDE_CHIPSETS) [N]
Loopback device support (CONFIG_BLK_DEV_LOOP) [N]
Network block device driver support (CONFIG_BLK_DEV_NBD) [N]
Multiple device driver support (CONFIG_BLK_DEV_MD) [N]
RAM disk support (CONFIG_BLK_DEV_RAM) [N]
XT hard disk support (CONFIG_BLK_DEV_XD) [N]
Mylex DAC960/DAC1100 PCI RAID Controller support (CONFIG_BLK_DEV_DAC960) [N]
Parallel port IDE device support (CONFIG_PARIDE) [N]
Compaq SMARTZ support (CONFIG_BLK_CPQ_DA) [N]

Networking options:
Packet socket (CONFIG_PACKET) [Y]
Kernel/user netlink socket (CONFIG_NETLINK) [N]
Network firewalls (CONFIG_FIREFALL) [N]
Socket filtering (CONFIG_FILTER) [N]
Unix domain sockets (CONFIG_UNIX) [Y]
TCP/IP networking (CONFIG_INET) [Y]
IP:Multicasting (CONFIG_IP_MULTICAST) [N]
IP:Advanced router (CONFIG_IP_ADVENCED_ROUTER) [N]
IP:Kernel level autoconfiguration (CONFIG_IP_PNP) [N]
IP:Firewalling (CONFIG_IP_FIREWALL) [N]
IP:Optimize as router not host (CONFIG_IP_ROUTER) [N]
IP:Tunneling (CONFIG_NET_IPIP) [N]
IP:GRE tunnels over IP (CONFIG_NET_IPGRE) [N]
IP:Aliasing support (CONFIG_IP_ALIAS) [N]
IP:TCP syncookie support (CONFIG_SYN_COOKIES) [N]
IP:Reverse ARP (CONFIG_INET_RARP) [N]
IP:Allow large windows (CONFIG_SKB_LARGE) [Y]
The IPX protocol (CONFIG_IPX) [N]
IPX: Full internal IPX network (CONFIG_IPX_INTERN) [N]
AppleTalk DDP (CONFIG_ATALK) [N]

QoS and/or fair queuering:
N/A

SCSI support:
SCSI support (GONFIG_SCSI) [Y]
SCSI disk support (CONFIG_BLK_DEV_SD) [Y]
SCSI tape support (CONFIG_CHR_DEV_ST) [N]
SCSI CD-ROM support (CONFIG_BLK_DEV_SR) [N]
SCSI generic support (CONFIG_CHR_DEV_SG) [N]
Probe all LUMs on each SCSI device (CONFIG_SCSI_MULTI-LUM) [Y] **N**
Verbose SCSI error reporting (kernel size +=12K) (CONFIG_SCSI_CONSTANTS) [Y] **N**
SCSI logging facility (CONFIG_SCSI_LOGGING) [N]

SCSI low-level drivers:
Adaptec AIC7xxx support (CONFIG_SCSI_AIC7XXX) [N] **Y**
Enable Tagged Command Queuering (CONFIG_AIC7XXX_TCQ_ON_BY_DEFAULT) [N] **Y**
SYM53C8XX SCSI support (CONFIG-SCSI_SYM53C8XX) [Y] **N**

Network device support:
Network device support (CONFIG_NETDEVICES) [Y]
Dummy net driver support (CONFIG_DUMMY) [M] **N**
EQL (serial line load balancing) support (CONFIG_EQUALIZER) [N]
General Instruments Surfboard 1000 (CONFIG_NET_SB1000) [N]
FDDI driver support (CONFIG_FDDI) [N]
PPP (point-to-point) support (CONFIG_PPP) [N]
SLIP (serial line) support (CONFIG_SLIP) [N]
Wireless LAN (non-hamradio) (CONFIG_NET_RADIO) [N]
Fibre Channel driver support (CONFIG_NET_FC) [N]

ARQnet devices:
ARCnet support (CONFIG_ARCNET) [N]

Ethernet (10 or 100Mbit):
Ethernet (10 or 100Mbit) (CONFIG_NET_ETHERNET) [Y]
EISA, VLB, PCI and on board controllers (CONDIF_NET_EISA) [Y]
EtherExpressPro/100 support (CONFIG_EEXPRESS_PRO100) [Y]

AppleTalk devices:
N/A

Token ring devices:
Token Ring driver support (CONFIG_TR) [N]

Wan interfaces:
Comtrol Hostess SV-11 support (CONFIG_HOSTESS_SV11) [N]
COSA/SRP sync serial boards support (CONFIG_COSA) [N]
Sealevel Systems 4021 support (CONFIG_SEALEVEL_4021) [N]
Frame relay DLCI support (CONFIG_DLCI) [N]
WAN drivers (CONFIG_WAN_DRIVERS) [N]

Amateur Radio support:
Amateur Radio support (CONFIG_HAMRADIO) [N]

IrDA subsystem support:
IrDA subsystem support (CONFIG_IRDA) [N]

Infrared-port device drivers:
N/A

ISDN subsystem:
ISDN support (CONFIG_ISDN) [N]

Old CD-ROM drivers (not SCSI, not IDE):
Support non-SCSI/IDE/ATAPI CDROM drives (CONFIG_CD_NO_IDESCSI) [N]

Character devices:
Virtual terminal (CONFIG_VT) [Y]
Support for console on virtual terminal (CONFIG_VT_CONSOLE) [Y]
Standard/generic (dumb) serial support (CONFIG_SERIAL) [Y]
Support for console on special port (CONFIG_SERIAL_CONSOLE) [N]
Extended dumb serial driver options (CONFIG_SERIAL_EXTENDED) [N]
Non-standard serial port support (CONFIG_SERIAL_NONSTANDAR) [N]
Unix98 PTY support (CONFIG_UNIX98_PTYS) [Y] **N**
Maximum numbers of Unix98 PTYs in use (0-2048) (CONFIG_UNIX98_PTY_COUNT) [256] **128**
Mouse support (Not serial mice) (CONFIG_MOUSE) [Y]
QIC-02 tape support (CONFIG_QIC02_TAPE) [N]
Watchdog Timer support (CONFIG_WATCHDOG) [N]
/dev/nvram support (CONFIG_NVRAM) [N]
Enhanced Real Time Clock support (CONFIG_RTC) [N]
Double Talk PC internal speech controller support (CONFIG_DTLK) [N]

Mice:
ATIXL busmouse support (CONFIG_ATIXL_BUSMOUSE) [N]
Logitech busmouse support (CONFIG_BUSMOUSE) [N]
Microsoft busmouse support (CONFIG_MS_BUSMOUSE) [N]
PS/2 mouse (aka “auxiliary device”) support (CONFIG_PSMOUSE) [Y]
C&T 82C710 mouse port support (CONFIG_82c710_MOUSE) [Y] **N**
PC110 digitizer pad support (CONFIG_PC110_PAD) [N]

Watchdog Cards:
N/A

Video for Linux:
Video for Linux (CONFIG_VIDEO_DEV) [N]

Joystick support:
Joystick support (CONFIG_JOYSTICK) [N]

Ftape, the floppy tape device driver:
Ftape (QIC-80/Travan) support (CONDFIG_FTAPE) [N]

Filesystems:
Quota support (CONFIG_QUOTA) [N]
Kernel automounter support (CONFIG_AUTOFS_FS) [Y] **N**
Amiga FFS filesystem support (CONFIG_AFFS_FS) [N]
Apple Macintosh filesystem support (CONFIG_HFS_FS) [N]
DOS FAT fs support (CONFIG_FAT_FS) [N]
ISO 9660 CDROM filesystem support (CONFIG_ISO9660FS) [Y]
Microsoft Joliet CDROM extensions (CONFIG_JOLIET) [N]
Minix fs support (CONFIG_MINIX_FS) [N]
NTFS filesystem support (CONFIG_NTFS_FS) [N]
OS/2 MPFS filesystem support (CONFIG_HPFS_FS) [N]
/proc filesystem support (CONFIG_PROC_FS) [Y]
/dev/pts filesystem support (CONFIG_DEVPTS_FS) [Y] **N**
ROM filesystem support (CONFIG_ROMFS_FS) [N]
Second extended filesystem (CONFIG_EXT2_FS) [Y]
System V and coherent filesystem support (CONFIG_SYSV_FS) [N]
UFS filesystem support (CONFIG_UFS_FS) [N]

Network File Systems:
Coda filesystem support (Advanced Network fs) (CONFIG_CODA_FS) [N]
NFS filesystem support (CONFIG_NFS_FS) [Y] **N**

SMB filesystem support (CONFIG_SMB_FS) [N]
NCP filesystem support (CONFIG_NCP_FS) [N]

Partition Types:
BSD disklabel (BSD partition tables) support (CONFIG_BSD_DISKLABEL) [N]
Macintosh partition map support (CONFIG_MAC_PARTION) [N]
SMD disklabel (Sun partition tables) support (CONFIG_SMD_DISKLABEL) [N]
Solaris (x86) partition table support (CONFIG_SOLARIS_X86_PARTITION) [N]

Native Language Support:
N/A

Console drivers:
VGA text console (CONFIG_VGA_CONSOLE) [Y]
Video mode selection support (CONFIG_VIDEO_SELECT) [N]

Sound:
Sound card support (CONFIG_SOUND) [N]

Additional low level sound drivers:
N/A

Kernel hacking:
Magic SysRq key (CONFIG_MAGIC_SYSRQ) [N]

Now, return to the */usr/src/linux/* directory (if you are not already on). You need to compile the new kernel. You do so by using the following command:

[root@deep]# **make dep; make clean; make bzImage**

This line contains three commands in one. The first one, **make dep**, actually takes your configuration and builds the corresponding dependency tree. This process determines what gets compiled and what doesn't. The next step, **make clean**, erase all previous traces of a compilation so as to avoid any mistakes in which version of a feature gets tied into the kernel. Finally, **make bzImage** does the full compilation. After the process is complete, the kernel is compressed and ready to be installed.

Before you can install the new kernel, you need to compile the corresponding modules. This is requiring only if you're saying yes to "Enable loadable module support (CONFIG_MODULES)" and are compiled some options as a module above. You do so by using the following command:

[root@deep]# **make modules** (only if you're say yes to Enable loadable module support (CONFIG_MODULES) above).
[root@deep]# **make modules_install** (only if you're say yes to Enable loadable module support (CONFIG_MODULES) above).

Copy the file **/usr/src/linux/arch/i386/boot/bzImage** from the kernel source tree to **/boot**, and give it an appropriate new name.

[root@deep]# **cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-kernel.version.number**

Copy the file **/usr/src/linux/System.map** from the kernel source tree to **/boot**, and give it an appropriate new name.

[root@deep]# **cp /usr/src/linux/System.map /boot/System.map-kernel.version.number**

Cd into the */boot* directory and rebuild the links **vmlinuz** and **System.map** with the following commands

[root@deep]# **cd /boot**
[root@deep]# **ln -fs vmlinuz-kernel.version.number vmlinuz**
[root@deep]# **ln -fs System.map-kernel.version.number System.map**

Remove obsolete files to make space.

[root@deep]# **rm -f module-info**
[root@deep]# **rm -f initrd-2.2.12-20.img** (if you have a SCSI drive)

Finally, you need to edit the ***/etc/lilo.conf*** file to make your new kernel one of the boot time options.

[root@deep]# vi /etc/lilo.conf and make the appropriated change.

For example:
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=20
restricted
password=somepasswd
image=/boot/**vmlinuz-kernel.version.number** #(add your new kernel name file here).
label=linux
root=/dev/sda6
read-only

[root@deep]# **/sbin/lilo -v** (to update the lilo.conf file with the new kernel)

[root@deep linux]# /sbin/lilo -v
LILO version 21, [Copyright 1992-1998 Werner Almesberger

Reading boot sector from /dev/sda
Merging with /boot/boot.b
Boot image: /boot/vmlinuz-2.2.13
Added linux *
/boot/boot.0800 exits – o backup copy made.
Writing boot sector.

IMPORTANT NOTE: if you are say **NO** to the option Unix98 PTY support (CONFIG_UNIX98_PTYS) listed above, edit your */etc/fstab* file and remove the line that read

none /dev/pts devpts gid=5,mode=620 0 0

[root@deep]# vi /etc/conf.modules and delete all no necessary module lines

Example of no necessary module lines after recompilation of the kernel.
Alias scsi_hostadapter model (if you have a SCSI drive).
Alias eth0 Ethernet model card (for Ethernet cards).
Alias parport_lowlevel parport-pc (for parallel port).
Pre_install pcmcia_core /etc/rc.d/init.d/pcmcia start (for laptop).

Since these modules are now unselected or incorporated in our new kernel, we do not need them as a module. Remember to rerun **LILO** after making changes. **Reboot** and then test your results.

Making a new rescue floppy

If all has gone well, you should have a system with an upgraded kernel. You should now make a new rescue image with this kernel in the case of future emergencies. You should follow the previous instructions, just changing the arguments to mkbootdisk to cover the new version of the kernel.

Login as root, and insert a new floppy.

[root@deep]# **mkbootdisk --device /dev/fd0 2.2.13**
Insert a disk in /dev/fd0. Any information on the disk will be lost.
Press <Enter> to continue or ^C to abort:

Update your /dev entries

If you've done a major kernel upgrade, or have added new devices to your system, you're sure to be well of if you update your */dev* entries. You can do that with the command (as root) **./MAKEDEV update**. You can also use MAKEDEV to create a standard batch of /dev entries or separate ones. See man MAKEDEV NOTE: MAKEDEV is a script that utilizes mknod.

[root@deep]# **cd /dev**
[root@deep]# **./MAKEDEV update**

XVI) Install more than one Ethernet Card per Machine

If the driver(s) is/are being used as a loadable module

In the case of PCI drivers, the module will typically detect all of the installed cards automatically. For ISA cards, you need to supply the I/O base address of the card so the module knows where to look. This information is stored in the file */etc/conf.modules*.

As an example, consider we have two ISA 3c509 cards, one at I/O 0x300 and one at I/O 0x320.
vi /etc/conf.modules and add

alias eth0 3c509
alias eth1 3c509
options 3c509 io=0x300,0x320

This says that the 3c509 driver should be loaded for either eth0 or eth1 (alias eth0, eth1) and it should be loaded with the options io=0x300,0x320 so that the drivers knows where to look for the cards. Note that 0x is important – things like 300h as commonly used in the DOS world won't work.

For PCI cards, you typically only need the alias lines to correlate the ethN interfaces with the appropriate driver name, since the I/O base of a PCI card can be safety detected.
Vi /etc/conf.modules and add

alias eth0 3c509
alias eth1 3c509

If the drivers(s) is/are compiled into the kernel

PCI probes will find all related cards automatically. ISA card will still need to do the following. This information is stored in the file */etc/lilo.conf*. The method is to pass boot-time arguments to the kernel, which is usually done by **LILO**.
vi /etc/lilo.conf and add

append=”ether=0,0,eth1”

In this case eth0 and eth1 will be assigned in the order that the cards are found at boot. Since we have recompiled the kernel, we must to use the second method (If the drivers(s) is/are compiled into the kernel) to install our second Ethernet card on the system.

XVII) Configuring TCP/IP Networking manually with the command line

Ifconfig is the tool used to set up and configure your network card. You should understand this command in the event you need to configure the network by hand.

To assigns the eth0 interface the IP-address of 209.164.186.11 use the command:
[root@deep]# **ifconfig eth0 209.164.186.11 netmask 255.255.255.0**

To display all interfaces you may have, use the command:
[root@deep]# **ifconfig** without any parameters.

The output should look something like this.

```
eth0    Link encap:Ethernet  HWaddr 00:E0:18:90:1B:56
        inet addr:209.164.186.11  Bcast:209.164.186.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:1295 errors:0 dropped:0 overruns:0 frame:0
        TX packets:1163 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:100
        Interrupt:11 Base address:0xa800

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        UP LOOPBACK RUNNING  MTU:3924  Metric:1
        RX packets:139 errors:0 dropped:0 overruns:0 frame:0
        TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
```

If ifconfig is invoked without any parameters, it displays all interfaces you configured. An option of -a show the inactive one as well.
[root@deep]# **ifconfig -a**

The output should look something like this.

```
eth0    Link encap:Ethernet  HWaddr 00:E0:18:90:1B:56
        inet addr:209.164.186.11  Bcast:209.164.186.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:1295 errors:0 dropped:0 overruns:0 frame:0
        TX packets:1163 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:100
        Interrupt:11 Base address:0xa800

eth1    Link encap:Ethernet  HWaddr 00:E0:18:90:1B:56
        inet addr:199.1.1.116  Bcast:199.1.1.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:1295 errors:0 dropped:0 overruns:0 frame:0
        TX packets:1163 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:100
        Interrupt:5 Base address:0xa320

lo      Link encap:Local Loopback
```

```
inet addr:127.0.0.1  Mask:255.0.0.0
UP LOOPBACK RUNNING  MTU:3924  Metric:1
RX packets:139 errors:0 dropped:0 overruns:0 frame:0
TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
```

NOTE When using *ifconfig* to configure your network devices, the settings will not survive a reboot.

To assign the default gateway for 209.164.186.1 use the command:
[root@deep]# **route add default gw 209.164.186.1**

In this example, the default route is set up to go to 209.164.186.1, your router.

Verify that you can rich your hosts. Choose a host from your network, for instance 209.164.186.19 and type:
[root@deep]# **ping 209.164.186.19**

The output should look something like this.

```
[root@dios2 networking]# ping 209.164.186.19
PING 209.164.186.19 (209.164.186.19) from 209.164.186.11 : 56 data bytes
64 bytes from 209.164.186.19: icmp_seq=0 ttl=128 time=1.0 ms
64 bytes from 209.164.186.19: icmp_seq=1 ttl=128 time=1.0 ms
64 bytes from 209.164.186.19: icmp_seq=2 ttl=128 time=1.0 ms
64 bytes from 209.164.186.19: icmp_seq=3 ttl=128 time=1.0 ms

--- 209.164.186.19 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.0/1.0/1.0 ms
```

You should now display the routing information with the command **route** to see if both hosts have the correct routing entry:
[root@deep]# **route -n**

The output should look something like this.

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.1.1	0.0.0.0	255.255.255.255	UH	0	0	0	eth0
192.168.1.0	192.168.1.1	255.255.255.0	UG	0	0	0	eth0
192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
127.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0	lo

To check the status of the interfaces quickly, use the netstat -i command, as follows:
[root@deep]# **netstat -i**

The output should look something like this.

Kernel Interface table

Iface	MTU	Met	RX-OK	RX-ERR	RX-DRP	RX-OVR	TX-OK	TX-ERR	TX-DRP	TX-OVR	Flg
eth0	1500	0	4236	0	0	0	3700	0	0	0	BRU
lo	3924	0	13300	0	0	0	13300	0	0	0	LRU
ppp0	1500	0	14	1	0	0	16	0	0	0	PRU

Another useful netstat option is -t, which shows all active TCP connections. Following is a typical result of netstat -t:
[root@deep]# **netstat -t**

The output should look something like this.

Active Internet connections (w/o servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
Tcp	0	0	deep.openar:netbios-ssn	gate.openarch.com:1045	ESTABLISHED

Tcp	0	0	localhost:1032	localhost:1033	ESTABLISHED
Tcp	0	0	localhost:1033	localhost:1032	ESTABLISHED
Tcp	0	0	localhost:1030	localhost:1034	ESTABLISHED
Tcp	0	0	localhost:1031	localhost:1030	ESTABLISHED
Tcp	0	0	localhost:1028	localhost:1029	ESTABLISHED
Tcp	0	0	localhost:1029	localhost:1028	ESTABLISHED
Tcp	0	0	localhost:1026	localhost:1027	ESTABLISHED
Tcp	0	0	localhost:1027	localhost:1026	ESTABLISHED
Tcp	0	0	localhost:1024	localhost:1025	ESTABLISHED
Tcp	0	0	localhost:1025	localhost:1024	ESTABLISHED

To stop all networks devices on your system, use the command:
[root@deep]# **/etc/rc.d/init.d/network stop**

To start all networks devices on your system, use the command:
[root@deep]# **/etc/rc.d/init.d/network start**

Files configurations for each network device you have on your system are located in the ***/etc/sysconfig/network-scripts/*** directory with Red Hat Linux 6.1 and are named **ifcfg-eth0** for the first interface and **ifcfg-eth1** for the second etc. This file is creating dynamically by the kernel when you add a new card on your system.

Edit **/etc/sysconfig/network-scripts/ifcfg-eth0** and the output should look something like this.

```
DEVICE=eth0
IPADDR=207.164.186.11
NETMASK=255.255.255.0
NETWORK=207.164.186.0
BROADCAST=207.164.186.255
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
```

If you want to modify your network address manually or add new network on new interface, edit this file (ifcfg-ethN) or create a new one and make the appropriated change.

The /etc/hosts file
As your machine gets started, it will need to know the mapping of some hostnames to IP addresses before DNS can be referenced. This mapping is kept in the /etc/hosts file.

Following is a sample */etc/hosts* file:

IP Address	Hostname	Alias
127.0.0.1	localhost	deep.openarch.com
192.168.1.1	deep.openarch.com	deep
192.168.1.2	forest.openarch.com	forest

The leftmost column is the IP address to be resolved. The next column is that host's name. Any subsequent columns are alias for that host. In the second line, for example, the address 192.168.1.1 if for the host deep.openarch.com. Another name for deep.openarch.com is deep.

IMPORTANT NOTE: The **tcpd** program is responsible to monitored incoming requests for **telnet**, **ftp** and other services that have a one-to-one mapping onto executable files.

Operation is as follows: whenever a request for service arrives, the inetd daemon is tricked into running the tcpd program instead of the desired server. tcpd logs the request and does some additional checks. When all is well, tcpd runs the appropriate server program and goes away.

tcpd verifies the client host name that is returned by the address->name DNS server by looking at the host name and address that are returned by the name->address DNS server. If any discrepancy is detected, tcpd concludes that it is dealing with a host that pretends to have someone else's host name. Optionally, tcpd disables source-routing socket options on every connection that it deals with. This will take care of most attacks from hosts that pretend to have an address that belongs to someone else's network. UDP services do not benefit from this protection. Time out problems are often caused by the server trying to resolve the client IP address to a DNS name. Either DNS isn't configured properly on your server or the client machines aren't known to DNS. If you are intended to run **telnet** or **ftp** services on your server, and aren't using DNS, don't forget to add client machine name and IP in your ***/etc/hosts*** file on the server or you can expect to wait several minutes for the DNS lookup to time out, before you get a login: prompt.

XVIII) Install software's

Linux DNS and BIND Server

Overview

DNS is the **MOST** important network service for IP networks. All Unix Client machines should be configured to perform *caching functions* at minimum. Setting up a caching server for Client local machines will reduce load on the site's primary server. A caching only name server will find the answer to name queries and remember the answer the next time we need it. This will shorten the waiting time the next time significantly. For security reason, it is very important that DNS doesn't exist between hosts on the corporate network and external hosts, it is far safer to simply use IP addresses to connect to external machines from the corporate network and vice-versa. In our configuration and installation we'll run BIND as non root-user and in a chrooted environment.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account "root".
Bind version number is 8.2.2

Packages

Bind Homepage: http://www.isc.org/
You must be sure to download: bind-contrib.tar.gz, bind-doc.tar.gz, bind-src.tar.gz

Tarballs

I would advise using tarballs (tar.gz), it is a good idea to make a list of files on the system before you install Bind, and one afterwards, and then compare them using '**diff**' to find out what file it placed where. Simply run '**find /* > filedns1.txt**' before and '**find /* > filedns2.txt**' after you install the software, and use '**diff filedns1.txt filedns2.txt > diffdns.txt**' to get a list of what changed.

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# mkdir /var/tmp/bind
[root@deep]# cp bind-contrib.tar.gz /var/tmp/bind/
[root@deep]# cp bind-doc.tar.gz /var/tmp/bind/
[root@deep]# cp bind-src.tar.gz /var/tmp/bind/
```

Cd into the new Bind dir (cd /var/tmp/bind) and decompress the files:

```
[root@deep]# tar xzpf bind-contrib.tar.gz
[root@deep]# tar xzpf bind-doc.tar.gz
[root@deep]# tar xzpf bind-src.tar.gz
```

Configure and Optimize

Edit the **Makefile.set** file (vi src/port/linux/Makefile.set) and add:

```
'CC=egcs -D_GNU_SOURCE'
'CDEBUG=-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions'
'DESTBIN=/usr/bin'
'DESTSBIN=/usr/sbin'
'DESTEXEC=/usr/sbin'
'DESTMAN=/usr/man'
'DESTHELP=/usr/lib'
'DESTETC=/etc'
'DESTRUN=/var/run'
'DESTLIB=/usr/lib/bind/lib'
'DESTINC=/usr/lib/bind/include'
'LEX=flex -8 -l'
'YACC=yacc -d'
'SYSLIBS=-lfl'
'INSTALL=install'
'MANDIR=man'
'MANROFF=cat'
'CATEXT=$$N'
'PS=ps p'
'AR=ar crus'
'RANLIB=:'
```

Compile and Optimize

Type the following commands on your terminal

```
[root@deep]# make -C src
[root@deep]# make clean all -C src SUBDIRS=../doc/man
[root@deep]# make install -C src
[root@deep]# make install -C src SUBDIRS=../doc/man

[root@deep]# strip /usr/bin/addr
[root@deep]# strip /usr/bin/dig
[root@deep]# strip /usr/bin/dnsquery
[root@deep]# strip /usr/bin/host
[root@deep]# strip /usr/bin/nslookup
[root@deep]# strip /usr/bin/nsupdate
[root@deep]# strip /usr/bin/mkservdb
[root@deep]# strip /usr/sbin/named
[root@deep]# strip /usr/sbin/named-xfer
[root@deep]# strip /usr/sbin/ndc
[root@deep]# strip /usr/sbin/dnskeygen
[root@deep]# strip /usr/sbin/irpd
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf bind/
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **named.conf** file from the floppy.tgz archive to */etc/* directory.
Copy **named/** directory from the floppy.tgz archive to */var/* directory.
Copy **named** script from the floppy.tgz archive to */etc/rc.d/init.d/* directory.
Make symbolic rc.d links for **named** with the command **chkconfig --add named**.

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/named.conf file

After compiling DNS, you need to set up a domain for your server - we'll be using openarch.com as an example domain, and assuming you are using 192.168.1.0. What we'll be doing is setting up your server as a primary without slave server DNS and restricting access to it. To do this, add the following lines to your */etc/named.conf* (this assumes you are using the current version of bind, 8.2):

```
options {
        directory "/var/named";
        dump-file "/var/named/named.dump.db";
        pid-file "/var/run/named.pid";
        check-names master fail;
        check-names slave warn;
        check-names response ignore;
        notify no;
        datasize 20m;
        allow-query { 192.168.1/24; 127.0.0.0/8; };
        transfer-format many-answers;
        transfers-per-ns 2;
        allow-transfer { 192.168.1/24; 127.0.0.0/8; };
        // query-source address * port 53;
};

// These files are not specific to any zone
zone "." in {
            type hint;
            file "db.cache";
};

zone "0.0.127.in-addr.arpa" in {
            type master;
            file "db.127.0.0";
};

// These are our primary zone files
zone "openarch.com" in {
```

```
        type master;
        file "primary/db.openarch";
};

zone "1.168.192.in-addr.arpa" in {
        type master;
        file "primary/db.192.168.1";
};

// These are our slave zone files
```

Configuration of the /var/named/db.127.0.0 file

After adding these lines to named.conf, create the following files in */var/named/:* db.127.0.0

```
; Revision History: April 22, 1999 - admin@deep.openarch.com
; Start of Authority (SOA) records.
$TTL 345600
@  IN  SOA  deep.openarch.com. admin.deep.openarch.com.  (
                        00          ; Serial
                        86400       ; Refresh
                        7200        ; Retry
                        2592000     ; Expire
                        345600 )    ; Minimum

; Name Server (NS) records.
     NS   deep.openarch.com.

; only One PTR record.
1     PTR  localhost.
```

Configuration of the /var/named/primary/db.192.168.1 file

And then in */var/named/primary/*: db.192.168.1

```
; Revision History: April 22, 1999 - admin@deep.openarch.com
; Start of Authority (SOA) records.
$TTL 345600
@  IN  SOA  deep.openarch.com. admin.deep.openarch.com.  (
                        00          ; Serial
                        86400       ; Refresh
                        7200        ; Retry
                        2592000     ; Expire
                        345600 )    ; Minimum

; Name Server (NS) records.
     NS   deep.openarch.com.

; Addresses Point to Canonical Names (PTR) for Reverse lookups
1  PTR     deep.openarch.com.
2  PTR     forest.openarch.com.
3  PTR     gate.openarch.com.
```

Configuration of the /var/named/primary/db.openarch

And finally in */var/named/primary/*: db.openarch

```
; Revision History: April 22, 1999 - admin@deep.openarch.com
; Start of Authority (SOA) records.
$TTL 345600
@  IN  SOA  deep.openarch.com. admin.deep.openarch.com.  (
                        00          ; Serial
                        86400       ; Refresh
```

```
                        7200        ; Retry
                        2592000     ; Expire
                        345600 )    ; Minimum

; Name Server (NS) records.
      NS   deep.openarch.com.

; Mail Exchange (MX) records.
      MX   0  deep.openarch.com.

; Address (A) records.
localhost               A       127.0.0.1
deep                    A       192.168.1.1
forest                  A       192.168.1.2
gate                    A       192.168.1.3

; Aliases in Canonical Name (CNAME) records.
www                     CNAME  deep.openarch.com.
```

Important: before starting your DNS server you must take a copy of *db.cache* file and copy this file in */var/named/*. The *db.cache* tells your server where the servers for the “root” zone are. Use the following command on another Unix computer in your organization to query a new *db.cache* file for your DNS Server or pick one from your Red Hat Linux CD-ROM source distribution:

```
[root@deep]# dig @.aroot-servers.net . ns > db.cache
```

Don’t forget to copy the db.cache file to */var/named/* on your server where you’re installing DNS server after retrieving it.

Configuration of the /etc/rc.d/init.d/named script file

Configure your */etc/rc.d/init.d/named* script file to start and stop DNS/BIND daemons Server.

Create the **named** script file (touch /etc/rc.d/init.d/named) and add:

```
#!/bin/sh
#
# named         This shell script takes care of starting and stopping
#               named (BIND DNS server).
#
# chkconfig: - 55 45
# description: named (BIND) is a Domain Name Server (DNS) \
# that is used to resolve host names to IP addresses.
# probe: true

# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

[ -f /usr/sbin/named ] || exit 0

[ -f /etc/named.conf ] || exit 0

RETVAL=0

# See how we were called.
```

```
case "$1" in
 start)
      # Start daemons.
      echo -n "Starting named: "
      daemon named
            RETVAL=$?
            [ $RETVAL -eq 0 ] && touch /var/lock/subsys/named
            echo
      ;;
 stop)
      # Stop daemons.
      echo -n "Shutting down named: "
      killproc named
            RETVAL=$?
            [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/named
      echo
      ;;
 status)
            /usr/sbin/ndc status
            exit $?
            ;;
 restart)
            $0 stop
            $0 start
            ;;
 reload)
            /usr/sbin/ndc reload
            exit $?
            ;;
 probe)
            # named knows how to reload intelligently; we don't want linuxconf
            # to offer to restart every time
            /usr/sbin/ndc reload >/dev/null 2>&1 || echo start
            exit 0
            ;;

 *)
      echo "Usage: named {start|stop|status|restart}"
      exit 1
esac

exit $RETVAL
```

Now, make this script executable and change its default permission:
[root@deep]# **chmod 700 /etc/rc.d/init.d/named**

Start your DNS Server with the following command:
[root@deep]# **/etc/rc.d/init.d/named start**

Securing BIND/DNS

Running BIND in a chroot jail

This part focuses on preventing BIND from being used as a point of break-in to the system hosting it. Since BIND performs a relatively large and complex function, the potential for bugs that affect security is rather high. In fact, there have been exploitable bugs that allowed a remote attacker to obtain root access to hosts running BIND.

To minimize this risk, BIND can be run **as a non-root user**, which will limit any damage to what can be done as a normal user with a local shell. Of course, allowing what amounts to an

anonymous guest account falls rather short of the security requirements for most DNS servers, so an additional step can be taken - that is, **running BIND in a chroot jail**.

The main benefit of a chroot jail is that the jail will limit the portion of the filesystem the daemon can see to the root directory of the jail. Additionally, since the jail only needs to support BIND, the programs available in the jail can be extremely limited. Most importantly, there is no need for setuid-root programs, which, given the right (or wrong...) bug, can be used to gain root access and break out of the jail.

NOTE: named program must be in a directory listed in your PATH environmental variable for this to work. If you're root, and indeed do have BIND installed; this should be the case. For the rest of the documentation, I'll assume the path of your original named is */usr/sbin/named*.

Find the shared library dependencies of named. These will need to be copied into the chroot jail later.

```
[root@deep]# ldd /usr/sbin/named

libc.so.6 => /lib/libc.so.6 (0x40017000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Make a note of the files listed above; you will need these later.

Step 1:
Add a new user id and a new group id for running named. This is important because running it as root defeats the purpose of the jail, and using a different user id that already exists on the system can allow your services to access each others' resources. Think multi-layer security.

These are sample user and group id numbers. Check */etc/passwd* and */etc/group* for a free uid/gid number. We'll use 53.

```
[root@deep]# groupadd -g 53 named
[root@deep]# useradd -g 53 -u 53 named
```

Step 2:
Set up the chroot environment. First, create the root directory of the jail. We've chosen /chroot*/named* because we want to put this on its own separate filesystem to prevent filesystem attacks. Early in our installation procedure we are create a special partition */chroot* for this purpose.

```
[root@deep]# /etc/rc.d/init.d/named stop
[root@deep]# mkdir -p /chroot/named
```

Next, create the rest of directories like the following:

```
[root@deep]# mkdir /chroot/named/dev
[root@deep]# mkdir /chroot/named/lib
[root@deep]# mkdir /chroot/named/etc
[root@deep]# mkdir -p /chroot/named/usr/sbin
[root@deep]# mkdir -p /chroot/named/var/run
```

Copy the main configuration file, the zone files, the named and named-xfer programs:

```
[root@deep]# cp /etc/named.conf /chroot/named/etc/
root@deep]# mkdir /chroot/named/var/named
root@deep]# cd /var/named ; cp -a . /chroot/named/var/named/
root@deep]# mknod /chroot/named/dev/null c 1 3
root@deep]# chmod 666 /chroot/named/dev/null
```

root@deep]# **cp /usr/sbin/named /chroot/named/usr/sbin/**
root@deep]# **cp /usr/sbin/named-xfer /chroot/named/usr/sbin/**

Copy the shared libraries identified above and lastly, run **ldconfig -r /chroot/named/** to create a cache for any libraries that may be in the chroot.

root@deep]# **cp /lib/libc.so.6 /chroot/named/lib/**
root@deep]# **cp /lib/ld-linux.so.2 /chroot/named/lib/**
root@deep]# **ldconfig -r /chroot/named/**

NOTE: The command **ldconfig** must be executed after named process have been started and not before.

Copy the localtime and nsswitch.conf file to the jail so that log entries are adjusted for your local timezone properly:

root@deep]# **cp /etc/localtime /chroot/named/etc/**
root@deep]# **cp /etc/nsswitch.conf /chroot/named/etc/**

Step 3:
Tell syslogd about the new chrooted service.

Normally, processes talk to syslogd through */dev/log*. As a result of the chroot jail, this won't be possible, so syslogd needs to be told to listen to */chroot/named/dev/log*. To do this, edit the syslog startup script to specify additional places to listen.

Edit the **syslog** script (vi /etc/rc.d/init.d/syslog) to change the line:

daemon syslogd -m 0
To read:
daemon syslogd -m 0 -a /chroot/named/dev/log

Step 4:
Edit the **named** script (vi /etc/rc.d/init.d/named) to change the line:

[-f /usr/sbin/named] || exit 0
To read:
[-f /chroot/named/usr/sbin/named] || exit 0

[-f /etc/named.conf] || exit 0
To read:
[-f /chroot/named/etc/named.conf] || exit 0

daemon named
To read:
daemon /chroot/named/usr/sbin/named -t /chroot/named/ -unamed -gnamed

-t tells named to start up using the new chroot environment,
-u specifies the user to run as,
-g specifies the group to run as.

Red Hat's init scripts' daemon() function doesn't allow alternate PID files to be specified, but that won't affect the operation of the start and stop actions of the named init scripts since they will be called from outside the chroot jail.

As of BIND 8.2 ndc became a binary file, which renders the shipped ndc useless in this setting. To fix it, the BIND package must be compiled from source.

Keep in mind that these settings only apply to ndc. If you also want a new named and named xfer, the original settings should be used.

To do this, in the top level BIND source directory.

For ndc:
root@deep]# **cp bind-src.tar.gz /vat/tmp**
root@deep]# **cd /var/tmp/**
root@deep]# **tar xzpf bind-src.tar.gz**
root@deep]# **cd src**
root@deep]# **cp port/linux/Makefile.set port/linux/Makefile.set-orig**

Edit the **Makefile.set** file (vi port/linux/Makefile.set) to make the changes listed bellow:

'CC=**egcs** -D_GNU_SOURCE'
'CDEBUG=**-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions'**
'DESTBIN=/usr/bin'
'DESTSBIN=/chroot/named/usr/sbin'
'DESTEXEC=/chroot/named/usr/sbin'
'DESTMAN=/usr/man'
'DESTHELP=/usr/lib'
'DESTETC=/etc'
'DESTRUN=/chroot/named/var/run'
'DESTLIB=/usr/lib/bind/lib'
'DESTINC=/usr/lib/bind/include'
'LEX=flex -8 -I'
'YACC=yacc -d'
'SYSLIBS=-lfl'
'INSTALL=install'
'MANDIR=man'
'MANROFF=cat'
'CATEXT=$$N'
'PS=ps p'
'AR=ar crus'
'RANLIB=:'

root@deep]# **make clean**
root@deep]# **make**
root@deep]# **cp bin/ndc/ndc /usr/sbin/**
[root@deep]# **strip /usr/sbin/ndc**

It is a good idea to also build a new named now, to ensure the same version is used for both named and ndc.

For named:
root@deep]# **cd /var/tmp/src**
root@deep]# **cp port/linux/Makefile.set-orig port/linux/Makefile.set**

Edit the **Makefile.set** file (vi port/linux/Makefile.set) to make the changes listed bellow:

'CC=**egcs** -D_GNU_SOURCE'
'CDEBUG=**-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions'**
'DESTBIN=/usr/bin'
'DESTSBIN=/usr/sbin'
'DESTEXEC=/usr/sbin'
'DESTMAN=/usr/man'
'DESTHELP=/usr/lib'
'DESTETC=/etc'
'DESTRUN=/var/run'
'DESTLIB=/usr/lib/bind/lib'

'DESTINC=/usr/lib/bind/include'
'LEX=flex -8 -I'
'YACC=yacc -d'
'SYSLIBS=-lfl'
'INSTALL=install'
'MANDIR=man'
'MANROFF=cat'
'CATEXT=$$N'
'PS=ps p'
'AR=ar crus'
'RANLIB=:'

root@deep]# **rm .settings**
root@deep]# **make clean**
root@deep]# **make**
root@deep]# **cp bin/named/named /chroot/named/usr/sbin**
root@deep]# **cp bin/named-xfer/named-xfer /chroot/named/usr/sbin**
[root@deep]# **strip /chroot/named/usr/sbin/named**
[root@deep]# **strip /chroot/named/usr/sbin/named-xfer**
[root@deep]# **rm -f /usr/sbin/named**
[root@deep]# **rm -f /usr/sbin/named-xfer**
[root@deep]# **rm -f /etc/named.conf**
[root@deep]# **rm -rf /var/named/**

Step 5:
Test the new chrooted configuration! Restart syslogd:

root@deep]# **/etc/rc.d/init.d/syslog stop**
root@deep]# **/etc/rc.d/init.d/syslog start**

Now, start the new chrooted BIND:

root@deep]# **/etc/rc.d/init.d/named start**

Make sure it's running as named and with the new arguments.

root@deep]# **ps auxw | grep named**
named 11446 0.0 1.2 2444 1580 ? S 23:09 0:00 /named/usr/sbin/named –t /named/ -unamed -gnamed

The first column should be “named”, which is the user-id named is running under. The end of the line should be “*named -t /named/ -unamed -gnamed”*.

Cleanup after work
[root@deep]# rm –rf /var/tmp/bind/

NOTE: For security reason, it is very important that DNS doesn't exist between hosts on the corporate network and external hosts, it is far safer to simply use IP addresses to connect to external machines from the corporate network and vice-versa.

Zone transfers
a) Restrict Zone Transfers:
Restricting zone transfers prevents:

- Others from taxing the name server.
- Hackers from listing the contents of the zones.

```
options {
        Allow-transfer { 192.168.1/24; };
};
```

This control which name servers can transfers any zones from this name server. Note on restricting zone transfers: Remember to restrict zone transfers from slave name servers, not just the primary master.

b) Restrict the queries that your name servers accept to:

- The address they should come from.
- The zones they should ask about.

```
options {
        Allow-query { 192.168.1/24; 127.0.0.1; };
};
```

This specifies which IP addresses are allowed to send queries to the server. In particular, people who run Internet firewalls may have a legitimate need to hide certain parts of their name space from most of the world but to make if available to a limited audience.

Further documentation

For more details, there are several man pages you can read:

```
$ man dnsdomainname (1)   - show the system's DNS domain name
$ man dnskeygen (1)           - generate public, private, and shared secret keys for DNS Security
$ man dnsquery (1)            - query domain name servers using resolver
$ man named (8)               - Internet domain name server (DNS)
```

DNS Administrative Tools

dig

The *db.cache* tells your server where the servers for the “root” zone are. It must be updated periodically. The root name servers do not change very often, but they do change. A good practice is to check your *db.cache* file every month or two. Use the following command to query a new *db.cache* file for your DNS Server:

[root@deep]# **dig @.aroot-servers.net . ns > db.cache**

-Where **@.aroot-servers.net** is the address of the root server for query the new db.cache file and **db.cache** file is the name of your new db.cache file.

Copy the db.cache file to */var/named/* after retrieving it.
[root@deep]# **cp db.cache /var/named/**

ndc

This command allows the system administrator to control the operation of a name server. If no command is given, ndc will prompt for commands until it reads EOF. Type ndc on your terminal and then help to see help on different command for this program.

[root@deep]# **ndc**
[root@deep]# **ndc help**

DNS Users Tools

dnsquery

The dnsquery program is a general interface to nameservers via BIND resolver library calls. The program supports queries to the nameserver with an opcode of QUERY.

dnsquery <*-n nameserver*> <*host*>

For example:
[root@deep]# **dnsquery -n localhost 192.168.1.2**

-Where <*-n nameserver*> is the nameserver to be used in the query. *nameservers* can appear as either Internet addresses of the form w.x.y.z or can appear as domain names. <*host*> is the name of the host (or domain) of interest.

host
The host program looks for information about Internet hosts. It gets this information from a set of interconnected servers that are spread across the country. By default, it simply converts between host names and Internet addresses.

host <*hostnames or hostnumbers*>

For example:
[root@deep]# **host www.openarch.com**
www.openarch.com is a nickname for deep.openarch.com
deep.openarch.com has address 192.168.1.1

-Where <*hostnames or hostnumbers*> is either host names (www.openarch.com) or host numbers (192.168.1.1).

Installed files

/usr/bin/addr
/usr/bin/dig
/usr/bin/dnsquery
/usr/bin/host
/usr/bin/nslookup
/usr/bin/nsupdate
/usr/bin/mkservdb
/usr/sbin/named
/usr/sbin/named-xfer
/usr/sbin/named-bootconf
/usr/sbin/ndc
/usr/sbin/dnskeygen
/usr/sbin/irpd
/usr/lib/nslookup.help
/usr/lib/bind/
/etc/rc.d/init.d/named (chkconfig --del named)
/usr/man/man1/dig.1
/usr/man/man1/dnsquery.1
/usr/man/man1/host.1
/usr/man/man1/dnskeygen.1
/usr/man/man3/hesiod.3
/usr/man/man3/inet_cidr.3
/usr/man/man3/tsig.3
/usr/man/man3/getaddrinfo.3
/usr/man/man3/resolver.3
/usr/man/man3/getnetent.3
/usr/man/man5/irs.conf.5
/usr/man/man5/resolver.5
/usr/man/man5/named.conf.5
/usr/man/man7/hostname.7
/usr/man/man8/named-xfer.8
/usr/man/man8/named.8
/usr/man/man8/ndc.8
/usr/man/man8/nslookup.8
/usr/man/man8/named-bootconf.8
/usr/man/man8/nsupdate.8

At this part of our documentation, all software-listed bellows are optional and depend of what you wan to install or doing on your server. What king of job your server will do and for which part of your network Intranet/Internet etc. In other part it will be interesting for you to replace Telnet program with Ssh for secure remote administration. Another interesting program is Tripwire that aids system administrators and users in monitoring a designated set of files for any changes.

Linux SSH1 Server

Overview

SSH is a truly seamless and secure replacement of old, insecure remote login programs such as rlogin or rsh. According to the official SSH (Secure Shell) site, SSH is the secure login program that revolutionized remote management of networks hosts over the Internet. It is a powerful, very easy-to-use program that uses strong cryptography for protecting all transmitted confidential data, including passwords, binary files, and administrative commands. In our configuration we are configured sshd1 or sshd2 to support tcp-wrappers (inetd super server) for more security. SSH2 was originally free but is now under a commercial license, it is recommended to use SSH1 (free) instead of SSH2 (commercial). We provide in our configuration the both versions.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
Ssh1 version number is 1.2.27

Packages

SSH1 Homepage: http://www.ssh.fi/
You must be sure to download: ssh-1.2.27.tar.gz

Tarballs

I would advise using tarballs (tar.gz), it is a good idea to make a list of files on the system before you install ssh1, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filessh1.txt**’ before and ‘**find /* > filessh11.txt**’ after you install the software, and use ‘**diff filessh1.txt filessh11.txt > diffssh.txt**’ to get a list of what changed.

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# cp ssh-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf ssh-version.tar.gz
```

Compile and Optimize

Cd into the new Ssh1 directory and type the following commands on your terminal:

```
CC="egcs" \
CFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--with-etcdir=/etc/ssh \
--without-idea \
--enable-warnings \
--with-login \
--without-rsh \
--with-libwrap \
```

```
--disable-client-port-forwardings \
--disable-server-x11-forwarding \
--disable-client-x11-forwarding \
--disable-suid-ssh

[root@deep]# make
[root@deep]# make install
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf ssh1-version/ ssh-version.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **sshd_config** file to */etc/ssh/* directory.
Copy **ssh_config** file to */etc/ssh/* directory.
Copy **ssh** file to */etc/pam.d/* directory.

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configure the /etc/ssh/ssh_config file

The configuration file for ssh1 (/etc/ssh/ssh_config) allows you to set options that modify the operation of the client programs. The files contain keyword-value pairs, one per line, with keywords being case insensitive. Here are the more important keywords; a complete listing is available in the man page for ssh1.

Edit the **ssh_config** file (vi /etc/ssh/ssh_config) and add:

```
# Site-wide defaults for various options

 Host *
   ForwardAgent no
   ForwardX11 no
   RhostsAuthentication no
   RhostsRSAAuthentication no
   RSAAuthentication no
   TISAuthentication no
   PasswordAuthentication yes
   FallBackToRsh no
   UseRsh no
   BatchMode no
   Compression yes
   StrictHostKeyChecking no
   IdentityFile ~/.ssh/identity
   Port 22
```

KeepAlive yes
Cipher blowfish
EscapeChar ~

Configure the /etc/ssh/sshd_config file

The configuration file for sshd1 (/etc/ssh/sshd_config) allows you to set options that modify the operation of the daemon. The files contain keyword-value pairs, one per line, with keywords being case insensitive. Here are the more important keywords; a complete listing is available in the man page for sshd1.

Edit the **sshd_config** file (vi /etc/ssh/sshd_config) and add:

```
# This is ssh server systemwide configuration file.

Port 22
ListenAddress 192.168.1.1
HostKey /etc/ssh/ssh_host_key
RandomSeed /etc/ssh/ssh_random_seed
ServerKeyBits 1024
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin no
IgnoreRhosts yes
StrictModes yes
QuietMode no
X11Forwarding yes
X11DisplayOffset 10
FascistLogging no
PrintMotd yes
KeepAlive yes
SyslogFacility DAEMON
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication no
PasswordAuthentication yes
PermitEmptyPasswords no
UseLogin no
AllowUsers admin
AllowHosts gate.openarch.com
```

Configure sshd1 to use tcp-wrappers inetd super server

Tcp-wrappers take cares to start and stop sshd1 server. Upon execution, inetd reads its configuration information from a configuration file which, by default, is */etc/inetd.conf*. There must be an entry for each field of the configuration file, with entries for each field separated by a tab or a space.

Edit the **inetd.conf** file (vi /etc/inetd.conf) and add the line:

```
ssh     stream  tcp     nowait  root    /usr/sbin/tcpd  sshd -i
```

The –i parameter is important since is specifies that sshd is being run from inetd.

Edit the **hosts.allow** file (vi /etc/hosts.allow) and add the line:

```
sshd: 192.168.1.3/255.255.255.0 gate.openarch.com
```

-Which mean client IP “192.168.1.3” with host name “gate.openarch.com” is allowed to ssh on the server. Hostname is important since we use PARANOID option in our hosts.deny config file.

These "daemon" strings (for tcp-wrappers) are in use by sshd1:
sshdfwd-X11 (if you want to allow/deny X11-forwarding).
sshdfwd-<port-number> (for tcp-forwarding).
sshdfwd-<port-name> (port-name defined in /etc/services. Used in tcp-forwarding).

Configuration of the /etc/pam.d/ssh file

Configure your */etc/pam.d/ssh* file to use pam authentication.

Create the **ssh** file (touch /etc/pam.d/ssh) and add:

```
#%PAM-1.0
auth        required    /lib/security/pam_pwdb.so shadow
auth        required    /lib/security/pam_nologin.so
account     required    /lib/security/pam_pwdb.so
password    required    /lib/security/pam_cracklib.so
password    required    /lib/security/pam_pwdb.so use_authtok nullok md5 shadow
session     required    /lib/security/pam_pwdb.s o
```

Further documentation

For more details, there are several man pages you can read:

```
$ man ssh-add1 (1)        - adds identities for the authentication agent
$ man ssh-agent1 (1)      - authentication agent
$ man ssh-keygen1 (1)     - authentication key pair generation
$ man ssh1 (1)            - secure shell client (remote login program)
$ man sshd1 (8)           - secure shell daemon
```

Per-User Configuration

[root@deep]# **su username**

Create private & public keys of local, by executing:
[root@deep]# **ssh-keygen1**

The result should look like the following example:

```
Initializing random number generator...
Generating p:  .............................++ (distance 430)
Generating q:  .......................++ (distance 456)
Computing the keys...
Testing the keys...
Key generation complete.
Enter file in which to save the key (/root/.ssh/identity):
Enter passphrase:
Enter the same passphrase again:
Your identification has been saved in /root/.ssh/identity.
Your public key is:
1024 37
14937757511251955533691120318477293862290049394715136511145806108870001764378494676831
29757784315853227236120610062314604405364871843677484233240919418480988907860997175244
46977589647127757030728779973708569993017043141563536333068888944038178461608592483844
59020215410275690305584653406336563558489976540218 1 root@deep.openarch.com
Your public key has been saved in /root/.ssh/identity.pub
```

SSH1 Users Tools

Ssh1

Ssh1 (Secure Shell) is a program for logging into a remote machine and executing commands in a remote machine. It is intended to replace rlogin and rsh, and provide secure encrypted communications between two untrusted hosts over an insecure network. X11 connections and arbitrary TCP/IP ports can also be forwarded over the secure channel.

ssh1 -l *<login_name> hostname*

<login_name> is the name you use to connect to ssh server and *hostname* is the address of your ssh server.

For example:
[root@deep]# **ssh1 -l admin www.openarch.com**
admin@deep.openarch.com's password:
Last login: Tue Oct 19 1999 18:13:00 -0400 from localhost
Welcome to www.openarch.com on Deepforest.

Installed files

/etc/ssh
/etc/ssh/ssh_host_key
/etc/ssh/ssh_host_key.pub
/etc/ssh/ssh_config
/etc/ssh/sshd_config
/root/.ssh
/root/.ssh/random_seed
/usr/bin/ssh1
/usr/bin/ssh
/usr/bin/scp1
/usr/bin/slogin
/usr/bin/ssh-keygen1
/usr/bin/ssh-keygen
/usr/bin/ssh-agent1
/usr/bin/ssh-agent
/usr/bin/ssh-add1
/usr/bin/ssh-add
/usr/bin/scp
/usr/bin/make-ssh-known-hosts1
/usr/bin/make-ssh-known-hosts
/usr/man/man1/ssh-agent1.1
/usr/man/man1/ssh-keygen1.1
/usr/man/man1/ssh-add1.1
/usr/man/man1/scp1.1
/usr/man/man1/slogin1.1
/usr/man/man1/slogin.1
/usr/man/man1/ssh1.1
/usr/man/man1/make-ssh-known-hosts1.1
/usr/man/man1/make-ssh-known-hosts.1
/usr/man/man8/sshd1.8
/usr/sbin/sshd1
/usr/sbin/sshd

Linux SSH2 Server

Overview

SSH is a truly seamless and secure replacement of old, insecure remote login programs such as rlogin or rsh. According to the official SSH (Secure Shell) site, SSH is the secure login program that revolutionized remote management of networks hosts over the Internet. It is a powerful, very easy-to-use program that uses strong cryptography for protecting all transmitted confidential data, including passwords, binary files, and administrative commands. In our configuration we are configured sshd1 or sshd2 to support tcp-wrappers (inetd super server) for security reason. SSH2 was originally free but is now under a commercial license, it is recommended to use SSH1 (free) instead of SSH2 (commercial). We provide in our configuration the both versions.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).

Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
Ssh2 version number is 2.0.13

Packages

SSH2 Homepage: http://www.ssh.fi/
You must be sure to download: ssh-2.0.13.tar.gz

Tarballs

I would advise using tarballs (tar.gz), it is a good idea to make a list of files on the system before you install ssh2, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filessh1.txt**’ before and ‘**find /* > filessh2.txt**’ after you install the software, and use ‘**diff filessh1.txt filessh2.txt > diffssh.txt**’ to get a list of what changed.

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# cp ssh-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf ssh-version.tar.gz
```

Compile and Optimize

Cd into the new Ssh2 directory and type the following commands on your terminal:

```
CC=”egcs” \
CFLAGS="-O6 –mcpu=pentiumpro –march=pentiumpro –fomit-frame-pointer –fno-exceptions" \
./configure \
–prefix=/usr \
–without-ssh-agent1-compat \
–disable-suid-ssh-signer \
--disable-tcp-port-forwarding \
--disable-X11-forwarding \
–enable-tcp-nodelay \
--with-libwrap

[root@deep]# make clean
[root@deep]# make
[root@deep]# make install
[root@deep]# rm -f /usr/bin/ssh-askpass
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf ssh2-version/ ssh-version.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we’ll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have

a specific directory and subdirectory in the tar compressed file “floppy.tgz” containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **sshd2_config** file to */etc/ssh2/* directory.
Copy **ssh2_config** file to */etc/ssh2/* directory.
Copy **ssh** file to */etc/pam.d/* directory.

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configure the /etc/ssh2/ssh2_config file

The configuration file for ssh2 (/etc/ssh2/ssh2_config) allows you to set options that modify the operation of the client programs. The files contain keyword-value pairs, one per line, with keywords being case insensitive. Here are the more important keywords; a complete listing is available in the man page for ssh2.

Edit the **ssh2_config** file (vi /etc/ssh2/ssh2_config) and add:

```
# ssh2_config
# SSH 2.0 Client Configuration File

*:
    Port                    22
    Ciphers                 AnyStdCipher
    Compression             yes
    IdentityFile            identification
    AuthorizationFile       authorization
    RandomSeedFile          random_seed
    VerboseMode             no
    ForwardAgent            no
    ForwardX11              no
    PasswordPrompt          "%U's password: "
    Ssh1Compatibility       no
    Ssh1AgentCompatibility  none
    NoDelay                 yes
    KeepAlive               yes
    QuietMode               no
```

Configure the /etc/ssh2/sshd2_config file

The configuration file for sshd2 (/etc/ssh2/sshd2_config) allows you to set options that modify the operation of the daemon. The files contain keyword-value pairs, one per line, with keywords being case insensitive. Here are the more important keywords; a complete listing is available in the man page for sshd2.

Edit the **sshd2_config** file (vi /etc/ssh2/sshs2_config) and add:

```
# sshd2_config
# SSH 2.0 Server Configuration File

*:
    Port                    22
    Listen                  Address192.168.1.1
    Ciphers                 AnyStdCipher
    IdentityFile            identification
    AuthorizationFile       authorization
    HostKeyFile             hostkey
```

```
PublicHostKeyFile          hostkey.pub
RandomSeedFile             random_seed
ForwardAgent               no
ForwardX11                 no
PasswordGuesses            3
MaxConnections             5
PermitRootLogin            no
AllowedAuthentications     publickey,password
RequiredAuthentications    publickey,password
ForcePTTYAllocation        no
VerboseMode                no
PrintMotd                  yes
CheckMail                  yes
UserConfigDirectory        "%D/.ssh2"
SyslogFacility             AUTH
Ssh1Compatibility          no
NoDelay                    yes
KeepAlive                  yes
RequireReverseMapping      yes
UserKnownHosts             yes
QuietMode                  no

# subsystem definitions

subsystem-sftp             sftp-server
```

Configure sshd2 to use tcp-wrappers inetd super server

Tcp-wrappers take cares to start and stop sshd2 server. Upon execution, inetd reads its configuration information from a configuration file which, by default, is */etc/inetd.conf*. There must be an entry for each field of the configuration file, with entries for each field separated by a tab or a space.

Edit the **inetd.conf** file (vi /etc/inetd.conf) and add the line:

```
ssh     stream  tcp     nowait  root    /usr/sbin/tcpd  sshd -i
```

The –i parameter is important since is specifies that sshd is being run from inetd.

Edit the **hosts.allow** file (vi /etc/hosts.allow) and add the line:

```
sshd: 192.168.1.3/255.255.255.0 gate.openarch.com
```

-Which mean client “192.168.1.3” with host name “gate.openarch.com” is allowed to ssh on the server. Hostname is important since we use PARANOID option in our hosts.deny config file.

These "daemon" strings (for tcp-wrappers) are in use by sshd2:

sshd, sshd2	(The name sshd2 was called with (usually "sshd")).
sshdfwd-X11	(if you want to allow/deny X11-forwarding).
sshdfwd-<port-number>	(for tcp-forwarding).
sshdfwd-<port-name>	(port-name defined in /etc/services. Used in tcp-forwarding).

Configuration of the /etc/pam.d/ssh file

Configure your */etc/pam.d/ssh* file to use pam authentication.

Create the **ssh** file (touch /etc/pam.d/ssh) and add:

```
#%PAM-1.0
```

```
auth        required    /lib/security/pam_pwdb.so shadow
auth        required    /lib/security/pam_nologin.so
account     required    /lib/security/pam_pwdb.so
password    required    /lib/security/pam_cracklib.so
password    required    /lib/security/pam_pwdb.so use_authtok nullok md5 shadow
session     required    /lib/security/pam_pwdb.so
```

Further documentation

For more details, there are several man pages you can read:

```
$ man ssh-add2 (1)        - adds identities for the authentication agent
$ man ssh-agent2 (1)      - authentication agent
$ man ssh-keygen2 (1)     - authentication key pair generation
$ man ssh2 (1)            - secure shell client (remote login program)
$ man sshd2 (8)           - secure shell daemon
```

Per-User Configuration

[root@deep]# su username

Create private & public keys of local, by executing:
[root@deep]# **ssh-keygen2**

Create an “identification” file in your “.ssh2” directory on local:
[root@deep]# **cd ~/.ssh2**
[root@deep]# **echo “IdKey id_dsa_1024_a” > identification**

NOTE: It’s optional to create an identification file on Remote.

Copy your public key of Local (**id_dsa_1024_a.pub**) to “.ssh2” directory of remote under the name, say, “**Local.pub**”.

Create an “**authorization**” file in your “.ssh2” directory on Remote:
[root@deep]# **touch authorization**

Add the following one line to “authorization”:
[root@deep]# **vi authorization**
key Local.pub

SSH2 Users Tools

ssh2

Ssh2 (Secure Shell) is a program for logging into a remote machine and executing commands in a remote machine. It is intended to replace rlogin and rsh, and provide secure encrypted communications between two untrusted hosts over an insecure network. X11 connections and arbitrary TCP/IP ports can also be forwarded over the secure channel.

ssh2 -l <*login_name*> *hostname*

<*login_name*> is the name you use to connect to ssh server and *hostname* is the address of your ssh server.

For example:
[root@deep]# **ssh2 -l admin www.openarch.com**
Passphrase for key "/home/admin/.ssh2/id_dsa_1024_a" with comment "1024-bit dsa, admin@deep.openarch.com, Tue Oct 19 1999 14:31:40 -0400":
admin's password:

```
Last login: Tue Oct 19 1999 18:13:00 -0400 from localhost
Welcome to www.openarch.com on Deepforest.
```

sftp2
sftp (Secure File Transfer) is a ftp-like client that can be used in file transfer over the network. sftp uses Ssh2 in data connections, so the file transport is secure. You must already be connected with ssh2 before use sftp2.

sftp2 *hostname*

- *hostname* is the name of the server you wan to sftp.

For example:

```
[root@deep]# sftp2 www.openarch.com
local path  : /home/admin
Passphrase for key "/home/admin/.ssh2/id_dsa_1024_a" with comment "1024-bit dsa,
admin@deep.openarch.com, Tue Oct 19 1999 14:31:40 -0400":
admin's password:
admin's password:
remote path : /home/admin
sftp>
```

Installed files

```
/etc/ssh2/
/usr/bin/ssh2
/usr/bin/scp2
/usr/bin/sftp2
/usr/bin/sftp-server2
/usr/bin/ssh-agent2
/usr/bin/ssh-keygen2
/usr/bin/ssh-signer2
/usr/bin/ssh-add2
/usr/bin/ssh
/usr/bin/ssh-agent
/usr/bin/ssh-add
/usr/bin/ssh-keygen
/usr/bin/scp
/usr/bin/sftp
/usr/bin/sftp-server
/usr/bin/ssh-signer
/usr/sbin/sshd2
/usr/sbin/sshd
/usr/man/man1/ssh2.1
/usr/man/man1/ssh-keygen2.1
/usr/man/man1/ssh-add2.1
/usr/man/man1/ssh-agent2.1
/usr/man/man1/scp2.1
/usr/man/man1/sftp2.1
/usr/man/man1/ssh.1
/usr/man/man1/ssh-add.1
/usr/man/man1/ssh-agent.1
/usr/man/man1/ssh-keygen.1
/usr/man/man1/scp.1
/usr/man/man1/sftp.1
/usr/man/man8/sshd2.8
/usr/man/man8/sshd.8
~/ userdir .ssh2
```

Linux OPENSSL

Overview
The OpenSSL Project is a collaborative effort to develop a robust, commercial-grade, fully featured, and Open Source toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols with full-strength cryptography. The project is managed by a worldwide community of volunteers that use the Internet to communicate, plan, and develop the OpenSSL toolkit and its related documentation.

These installation instructions assume
Commands are Unix-compatible.

The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
OpenSSL version number is 0.9.4

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install Openssl, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filessl1.txt**’ before and ‘**find /* > filessl2.txt**’ after you install the software, and use ‘**diff filessl1.txt filessl2.txt > diffssl.txt**’ to get a list of what changed.

Packages

OpenSSL Homepage: http://www.openssl.org/
You must be sure to download: openssl-0.9.4.tar.gz

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# cp openssl_version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf openssl_version.tar.gz
```

Compile and Optimize

Cd into the new Openssl directory and type the following commands on your terminal:

[root@deep]# vi +11 tools/c_rehash and change:

```
DIR=/usr/local/ssl
To
DIR=/usr
```

[root@deep]# **perl util/perlpath.pl /usr/bin** (where your perl program reside).

[root@deep]# **export LD_LIBRARY_PATH=`pwd`**

```
CC=”egcs” \
./Configure linux-elf –DSSL_FORBID_ENULL \
--prefix=/usr  \
--openssldir=/etc/ssl
```

NOTE: “-DSSL_FORBID_ENULL” is requiring for not allowing null encryption.

[root@deep]# vi +52 Makefile.ssl and add:

CFLAG= -DTHREADS -D_REENTRANT –DSSL_FORBID_ENULL -DL_ENDIAN –DTERMIO **-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions** -Wall -DSHA1_ASM -DMD5_ASM -DRMD160_ASM
PROCESSOR= **686**

```
[root@deep]# make -f Makefile
[root@deep]# make test
[root@deep]# make install
```

[root@deep]# mv /etc/ssl/misc/* /usr/bin/

```
[root@deep]# rm -rf  /etc/ssl/misc/
[root@deep]# rm -rf  /etc/ssl/lib/
[root@deep]# rm -f  /usr/bin/CA.pl
[root@deep]# rm -f  /usr/bin/CA.sh
[root@deep]# install -m 644 libRSAglue.a  /usr/lib/
[root@deep]# install -m 644 rsaref/rsaref.h  /usr/include/openssl/
[root@deep]# strip /usr/bin/openssl
[root@deep]# mkdir -p /etc/ssl/crl
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm –rf openssl-version/ openssl_version.tar.gz
```

Configuration:

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **openssl.cnf** file to */etc/ssl/* directory.
Copy **sign.sh** script to */usr/bin/* directory.

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/ssl/openssl.cnf file

This is the general configuration file for openssl program where you can configure, expiration date of your keys, the name of your organization, the address etc.

Edit the **openssl.cnf** file (vi /etc/ssl/openssl.cnf) and add:

```
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#

RANDFILE                = $ENV::HOME/.rnd
oid_file                = $ENV::HOME/.oid
oid_section             = new_oids

# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions            =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)

[ new_oids ]

# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
```

```
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6

####################################################################
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = /etc/ssl              # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/ca.db.index      # database index file.
new_certs_dir   = $dir/ca.db.certs      # default place for new certs.

certificate     = $dir/certs/ca.crt     # The CA certificate
serial          = $dir/ca.db.serial     # The current serial number
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/ca.key   # The private key
RANDFILE        = $dir/ca.db.rand       # private random number file

x509_extensions = usr_cert              # The extentions to add to the cert

# Extensions to add to a CRL. Note: Netscape communicator chokes on V2 CRLs
# so this is commented out by default to leave a V1 CRL.
# crl_extensions = crl_ext

default_days    = 365                   # how long to certify for
default_crl_days= 30                    # how long before next CRL
default_md      = md5                   # which md to use.
Preserve        = no                    # keep passed DN ordering

# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy          = policy_match

# For the CA policy
[ policy_match ]
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

####################################################################
[ req ]
default_bits            = 1024
```

```
default_keyfile                 = privkey.pem
distinguished_name              = req_distinguished_name
attributes                      = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = CA
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Quebec

localityName                    = Locality Name (eg, city)
localityName_default            = Montreal

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Open Network Architecture

# we can do this but it is not needed normally :-)
#1.organizationName             = Second Organization Name (eg, company)
#1.organizationName_default     = World Wide Web Pty Ltd

organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = Internet Department

commonName                      = Common Name (eg, YOUR name)
commonName_default              = www.openarch.com
commonName_max                  = 64

emailAddress                    = Email Address
emailAddress_default            = admin@openarch.com
emailAddress_max                = 40

# SET-ex3                       = SET extension number 3

[ req_attributes ]
challengePassword               = A challenge password
challengePassword_min           = 4
challengePassword_max           = 20

unstructuredName                = An optional company name

[ usr_cert ]

# These extensions are added when 'ca' signs a request.

# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.

basicConstraints=CA:FALSE

# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.

# This is OK for an SSL server.
# nsCertType                    = server

# For an object signing certificate this would be used.
# nsCertType = objsign
```

```
# For normal client use this is typical
# nsCertType = client, email

# and for everything including object signing:
# nsCertType = client, email, objsign

# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment

# This will be displayed in Netscape's comment listbox.
nsComment                       = "OpenSSL Generated Certificate"

# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always

# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy

# Copy subject details
# issuerAltName=issuer:copy

#nsCaRevocationUrl              = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName

[ v3_ca]

# Extensions for a typical CA


# PKIX recommendation.

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer:always

# This is what PKIX recommends but some broken software chokes on critical
# extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true

# Key usage: this is typical for a CA certificate. However since it will
# prevent it being used as an test self-signed certificate it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign

# Some might want this also
# nsCertType = sslCA, emailCA

# Include email address in subject alt name: another PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy

# RAW DER hex encoding of an extension: beware experts only!
# 1.2.3.5=RAW:02:03
```

```
# You can even override a supported extension:
# basicConstraints= critical, RAW:30:03:01:01:FF

[ crl_ext ]
# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.

# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

Create the /usr/bin/sign.sh program file

The ``openssl ca'' commands has some strange requirements and the default OpenSSL config doesn't allow one easily to use ``openssl ca'' directly. So we'll create this "sign.sh" program to replace it.

Create the **sign.sh** program file (touch /usr/bin/sign.sh) and add:

```
#!/bin/sh
##
##  sign.sh -- Sign a SSL Certificate Request (CSR)
##  Copyright (c) 1998-1999 Ralf S. Engelschall, All Rights Reserved.
##

#   argument line handling
CSR=$1
if [ $# -ne 1 ]; then
   echo "Usage: sign.sign <whatever>.csr"; exit 1
fi
if [ ! -f $CSR ]; then
   echo "CSR not found: $CSR"; exit 1
fi
case $CSR in
  *.csr ) CERT="`echo $CSR | sed -e 's/\.csr/.crt/'`" ;;
     * ) CERT="$CSR.crt" ;;
esac

#   make sure environment exists
if [ ! -d ca.db.certs ]; then
   mkdir ca.db.certs
fi
if [ ! -f ca.db.serial ]; then
   echo '01' >ca.db.serial
fi
if [ ! -f ca.db.index ]; then
   cp /dev/null ca.db.index
fi

#   create an own SSLeay config
cat >ca.config <<EOT
[ ca ]
default_ca              = CA_own
[ CA_own ]
dir                     = /etc/ssl
certs                   = /etc/ssl/certs
new_certs_dir           = /etc/ssl/ca.db.certs
database                = /etc/ssl/ca.db.index
serial                  = /etc/ssl/ca.db.serial
RANDFILE                = /etc/ssl/ca.db.rand
certificate             = /etc/ssl/certs/ca.crt
private_key             = /etc/ssl/private/ca.key
```

```
default_days            = 365
default_crl_days        = 30
default_md              = md5
preserve                = no
policy                  = policy_anything
[ policy_anything ]
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional
EOT

#  sign the certificate
echo "CA signing: $CSR -> $CERT:"
openssl ca -config ca.config -out $CERT -infiles $CSR
echo "CA verifying: $CERT <-> CA cert"
openssl verify -CAfile /etc/ssl/certs/ca.crt $CERT

#  cleanup after SSLeay
rm -f ca.config
rm -f ca.db.serial.old
rm -f ca.db.index.old

#  die gracefully
exit 0
```

Now, make this program executable and change its default permission:
[root@deep]# **chmod 755 /usr/bin/sign.sh**

NOTE: you can also find this program “sign.sh” in the mod_ssl distribution *under mod_ssl-version/pkg.contrib/* subdirectory or on our floppy.tgz archive file.

Commands

The commands listed bellow are some that we use often in our regular use but much more exist and you must check the man page for more details and information.

NOTE: All command listed bellow are assumed to be made in ***/etc/ssl/*** directory.

1.1 Create a RSA private key protected with a passphrase for your Apache Server.

[root@deep]# **openssl genrsa -des3 -out server.key 1024**

Please backup this **server.key** file and remember the pass-phrase you had to enter at a secure location.

1.2 Generate a Certificate Signing Request (CSR) with the server RSA private key.

[root@deep]# **openssl req -new -key server.key -out server.csr**

You now have to send this Certificate Signing Request (CSR) to a Certifying Authority (CA) for signing. The result is then a real Certificate, which can be used for Apache. Here you have to options: First you can let the CSR sign by a commercial CA like Verisign or Thawte. Then you usually have to post the CSR into a web form, pay for the signing and await the signed Certificate you then can store into a server.crt file. Second you can use your own CA and now have to sign the CSR yourself by this CA. See bellow on how to sign a CSR with your CA yourself.

Make sure you enter the FQDN “Fully Qualified Domain Name” of the server when OpenSSL prompts you for the “CommonName”, i.e. when you generate a CSR for a website which will be later accessed via https://www.mydomain.com/, enter www.mydomain.com here.

1.3 Create a RSA private key for your ca (CA).

```
[root@deep]# openssl genrsa -des3 -out ca.key 1024
```

Please backup this **ca.key** file and remember the pass-phrase you had to enter at a secure location.

1.4 Create a self-signed (CA) certificate (x509 structure) with the RSA key of the CA.

```
[root@deep]# openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

```
[root@deep]# mv server.key private/
[root@deep]# mv ca.key private/
[root@deep]# mv ca.crt certs/
```

NOTE: The req command create a self-signed certificate when the –x509 switch is used.

1.5 Signing a certificate request. (We create and use our own Certificate Authority (CA))

Prepare a script for signing which is needed because the ``openssl ca'' command has some strange requirements and the default OpenSSL config doesn't allow one easily to use ``openssl ca'' directly. So a script named **sign.sh** is distributed with the floppy disk under openssl directory. Use this script for signing.

Now you can use this CA to sign server CSR's in order to create real SSL Certificates for use inside an Apache webserver (assuming you already have a server.csr at hand):

```
[root@deep]# /usr/bin/sign.sh server.csr
```

This signs the CSR and results in a server.crt file.
[root@deep]# **mv server.crt certs/**

Now you have two files: **server.key** and **server.crt**. These now can be used as following inside your Apache's **httpd.conf** file:

```
SSLCertificateFile    /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
```

The **server.csr** file is no longer needed.
[root@deep]# **rm -f server.csr**

Securing Openssl

```
[root@deep]# chmod 600 /etc/ssl/certs/ca.crt
[root@deep]# chmod 600 /etc/ssl/certs/server.crt
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/private/ca.key
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/private/server.key
```

Installed files

/etc/ssl/ /usr/bin/c_rehash

/usr/bin/CA.pl
/usr/bin/CA.sh
/usr/bin/c_hash
/usr/bin/c_info
/usr/bin/c_issuer
/usr/bin/c_name
/usr/bin/der_chop
/usr/bin/sign.sh
/usr/bin/openssl
/usr/lib/libRSAglue.a
/usr/lib/libcrypto.a
/usr/lib/libssl.a
/usr/include/openssl/

Linux Imap & Pop Server

Overview

POP stands for “Post Office Protocol” and simply allows you to list messages, retrieve them, and delete them. There are many POP servers for Linux available, the stock one that ships with most distributions is ok for the majority of users. IMAP is POP on steroids. It allows you to easily maintain multiple accounts, have multiple people access one account, leave mail on the server, just download the headers, or bodies and no attachments, and so on. IMAP is ideal for anyone on the go or with serious email needs. The default POP and IMAP servers that most distributions ship (bundled together into a single package named imapd oddly enough) fulfill most needs.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
Imap version number is 4.5

Packages

Imap Homepage: http://www.washington.edu/imap/
You must be sure to download: imap-4.5.tar.Z

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install Imap, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > fileimap1.txt**’ before and ‘**find /* > fileimap2.txt**’ after you install the software, and use ‘**diff fileimap1.txt fileimap2.txt > diffimap.txt**’ to get a list of what changed.

Compilation

Decompress the tarball (tar.Z).

```
[root@deep]# cp imap-version.tar.Z /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf imap-version.tar.Z
```

Compile and Optimize

Cd into the new Imap directory and type the following commands on your terminal:

```
[root@deep]# vi +698 src/osdep/unix/Makefile
```

Change in line 698: sh -c '(test -f /usr/include/sys/statvfs.h -a $(OS) != sc5 -a $(OS) != sco) && $(LN) flocksun.c flockbsd.c || $(LN) flocksv4.c flockbsd.c'
To: **sh -c '(test -f /usr/include/sys/statvfs.h -a $(OS) != sc5 -a $(OS) != sco -a $(OS) != lnx) && $(LN) flocksun.c flockbsd.c || $(LN) flocksv4.c flockbsd.c'**

[root@deep]# vi +355 src/osdep/unix/Makefile

Change in line 355: BASECFLAGS="-g -fno-omit-frame-pointer -O6 -DNFSKLUDGE" \
To: **BASECFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions -DNFSKLUDGE" **

[root@deep]# vi +112 src/osdep/unix/Makefile

Change in line 112: BUILDOPTIONS= EXTRACFLAGS="$(EXTRACFLAGS)"\
To: **BUILDOPTIONS= EXTRACFLAGS= -DDISABLE_POP_PROXY=1 -DIGNORE_LOCK_EACCES_ERRORS=1"$(EXTRACFLAGS)"**

[root@deep]# vi +58 src/osdep/unix/Makefile

Change in line 58: ACTIVEFILE=/usr/lib/news/active
To: **ACTIVEFILE=/var/lib/news/active**

Change: SPOOLDIR=/usr/spool
To: **SPOOLDIR=/var/spool**

Change: RSHPATH=/usr/ucb/rsh
To: **RSHPATH=/usr/bin/rsh**

vi +85 src/osdep/unix/Makefile

Change in line 85: CC=cc
To: **CC=egcs**

[root@deep]# make lnp

[root@deep]# mv ./c-client/c-client.a ./c-client/libimap.a
[root@deep]# install -m 644 c-client/libimap.a /usr/lib/
[root@deep]# install -m 644 ./src/ipopd/ipopd.8c /usr/man/man8/ipopd.8c
[root@deep]# install -m 644 ./src/imapd/imapd.8c /usr/man/man8/imapd.8c
[root@deep]# install -s -m 755 ./ipopd/ipop2d /usr/sbin/
[root@deep]# install -s -m 755 ./ipopd/ipop3d /usr/sbin/
[root@deep]# install -s -m 755 ./imapd/imapd /usr/sbin/
[root@deep]# mkdir -p /usr/include/imap
[root@deep]# install -m 644 ./c-client/*.h /usr/include/imap/
[root@deep]# install -m 644 ./src/osdep/tops-20/shortsym.h /usr/include/imap/
[root@deep]# chown root.mail /usr/sbin/ipop2d
[root@deep]# chown root.mail /usr/sbin/ipop3d
[root@deep]# chown root.mail /usr/sbin/imapd

NOTE: if you use only imap service remove pop2d and pop3d from your server. The same apply for popd, if you use only popd service remove imapd from your server.

Cleanup after work
[root@deep]# cd /var/tmp
[root@deep]# rm -rf imap-version/ imap-version.tar.Z

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and

other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **imap** file to */etc/pam.d/* directory.
Copy **pop** file to */etc/pam.d/* directory.

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/pam.d/imap file

Configure your */etc/pam.d/imap* file to use pam authentication.

Create the **imap** file (touch /etc/pam.d/imap) and add:

```
#%PAM-1.0
auth            required    /lib/security/pam_pwdb.so shadow nullok
account         required    /lib/security/pam_pwdb.so
```

Configuration of the /etc/pam.d/pop file

Configure your */etc/pam.d/pop* file to use pam authentication.

Create the **pop** file (touch /etc/pam.d/pop) and add:

```
#%PAM-1.0
auth            required    /lib/security/pam_pwdb.so shadow nullok
account         required    /lib/security/pam_pwdb.so
```

Further documentation

For more details, there are several man pages you can read:

```
$ man IMAPd (8c)          - Internet Message Access Protocol server
```

Installed files

```
/usr/sbin/ipop2d
/usr/sbin/ipop3d
/usr/sbin/imapd
/usr/lib/libimap.a
/usr/include/imap/
/etc/pam.d/imap
/etc/pam.d/pop
/usr/man/man8/ipopd.8c
/usr/man/man8/imapd.8c
```

Overview

Build the MM Shared Memory library when you want shared memory support in Apache/EAPI. For instance this allows mod_ssl to use a high-performance RAM-based session cache instead of a disk-based one.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
Mm version number is 1.0.12

Packages

MM Homepage: http://www.engelschall.com/sw/mm/
You must be sure to download: mm-1.0.12.tar.gz

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install MM, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filemm1.txt**’ before and ‘**find /* > filemm2.txt**’ after you install the software, and use ‘**diff filemm1.txt filemm2.txt > diffmm.txt**’ to get a list of what changed.

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# cp mm_version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf mm_version.tar.gz
```

Compile

Cd into the new mm directory and type the following commands on your terminal:

```
./configure \
--disable-shared  \
--prefix=/usr

[root@deep]# make
[root@deep]# make test
[root@deep]# make install
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf mm-version/ mm_version.tar.gz
```

Further documentation

For more details, there are several man pages you can read:

MM (3)	- Shared Memory Library
mm-config (1)	- MM library configuration/build utility

Installed files

/usr/bin/mm-config
/usr/include/mm.h
/usr/lib/libmm.la
/usr/lib/libmm.a
/usr/man/man1/mm-config.1
/usr/man/man3/mm.3

Linux Samba Server

Overview

Samba is the protocol by which a lot of PC-related machines share files and printers and other information such as lists of available files and printers. Operating systems that support this natively include Windows NT, OS/2, and Linux and add on packages that achieve the same thing are available for DOS, Windows, VMS, Unix of all kinds, MVS, and more. Apple Macs and some Web Browsers can speak this protocol as well. Alternatives to SMB include Netware, NFS, AppleTalk, Banyan Vines, Decnet etc; many of these have advantages but none are both public specifications and widely implemented in desktop machines by default.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
Samba version number is 2.0.5a

Packages

Samba Homepage: http://us1.samba.org/samba/samba.html
You must be sure to download: samba-2.0.5a.tar.gz

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install Samba, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filesmb1.txt**’ before and ‘**find /* > filesmb2.txt**’ after you install the software, and use ‘**diff filesmb1.txt filesmb2.txt > diffsmb.txt**’ to get a list of what changed.

Compilation

NOTE: These installations assume that you are running Shadow passwords/pam. (You really should be!).

Decompress the tarball (tar.gz).

```
[root@deep]# cp samba.version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf samba.version.tar.gz
```

Configure

Cd into the new Samba directory and then cd “sources” subdirectory.

[root@deep]# vi smbwrapper/smbsh.in and change:

```
SMBW_LIBDIR=${SMBW_LIBDIR-@builddir@/smbwrapper}
To:
SMBW_LIBDIR=${SMBW_LIBDIR-/usr/bin}
```

[root@deep]# vi Makefile.in and change:

```
SBINDIR  =  @bindir@
To:
SBINDIR  =  @sbindir@

VARDIR  =  @localstadir@
To:
VARDIR  =  /var/log/samba
```

[root@deep]# vi +10 script/convert_smbpasswd and change:

```
nawk 'BEGIN {FS=":"}
To:
gawk 'BEGIN {FS=":"}
```

[root@deep]# vi +650 include/include.h and remove:

```
Remove the lines:    #ifdef strcat
                     #undef strcat
                     #endif /* strcat */
                     #define strcat(dest,src) __ERROR__XX__NEVER_USE_STRCAT___;
```

[root@deep]# vi +96 client/smbmount.c and change:

```
static void close_our_files(int client_fd)
{
        int i;
        for (i = 0; i < 256; i++) {
                if (i == client_fd) continue;
                close(i);
    }
To:
static void close_our_files(int client_fd)
{
        struct rlimit limits;
        int i;

        getrlimit(RLIMIT_NOFILE,&limits);
        for (i = 0; i < limits.rlim_max; i++) {
                if (i == client_fd) continue;
                close(i);
        }
```

Compile and optimize

Type the following commands on your terminal:

```
CC="egcs" \
```

```
./configure \
--prefix=/usr \
--libdir=/etc \
--with-lockdir=/var/lock/samba \
--with-privatedir=/etc \
--with-swatdir=/usr/share/swat \
--with-pam

[root@deep]# make all
[root@deep]# make install

[root@deep]# install -m 755 script/mksmbpasswd.sh  /usr/bin/
[root@deep]# rm -rf /usr/share/swat/ (if like me, you don't like to configure samba in HTML)
[root@deep]# rm -f /usr/sbin/swat
[root@deep]# rm -f /usr/man/man8/swat.8
[root@deep]# mkdir -p /var/lock/samba
[root@deep]# mkdir -p /var/spool/samba
[root@deep]# chmod 1777 /var/spool/samba/
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf samba-version/ samba.version.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **smb.conf** and **lmhosts** files in */etc/* directory.
Copy **smb** script to */etc/rc.d/init.d/* directory.
Copy **samba** file to */etc/logrotate.d/* directory.
Copy **samba** file to */etc/pam.d/* directory.
Make symbolic rc.d links for **smb** with the command **chkconfig --add smb**.
Start your Samba Server with the following command

```
[root@deep]# /etc/rc.d/init.d/smb start
```

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/smb.conf file

The */etc/smb.conf* file is the main configuration file for Samba server, you can specify which directory you wan to access from windows machine, which IP addresses are authorized and so on. In our example we are created just one directory "/tmp" and are allowed one machine IP address to connect on the Samba server. We don't use printer capability over Samba and Windows.

Edit the **smb.conf** file (vi /etc/smb.conf) and add:

```
[global]
```

```
        workgroup = OPENARCH
        server string = Samba Server
        encrypt passwords = Yes
        smb passwd file = /etc/smbpasswd
        log file = /var/log/samba/log.%m
        max log size = 50
        security = user
        socket options = TCP_NODELAY
        domain master = Yes
        local master = Yes
        preferred master = Yes
        os level = 65
        wins support = Yes
        dns proxy = No
        name resolve order = wins lmhosts bcast host
        interfaces = 192.168.1.1/24 127.0.0.0/8
        bind interfaces only = Yes
        hosts allow = 192.168.1.3 192.168.1.1 127.
        hosts deny = ALL
        debug level = 1

[homes]
        comment = Home Directories
        read only = No
        browseable = No
        valid users = admin
        invalid users = root bin daemon nobody named sys tty disk mem kmem users

[tmp]
        comment = Temporary File Space
        path = /tmp
        read only = No
        valid users = admin
        invalid users = root bin daemon nobody named sys tty disk mem kmem users
```

Configuration of the /etc/lmhosts file

Configure your */etc/lmhosts* file.

Create the **lmhosts** file (touch /etc/lmhosts) and add:

```
# Sample Samba lmhosts file.
#
127.0.0.1       localhost
192.168.1.1     deep
192.168.1.3     gate
```

Configuration of the /etc/rc.d/init.d/smb script file

Configure your */etc/rc.d/init.d/smb* script file to start and stop Samba smbd and nmbd daemons Server.

Create the **smb** script file (touch /etc/rc.d/init.d/smb) and add:

```
#!/bin/sh
#
# chkconfig: - 91 35
# description: Starts and stops the Samba smbd and nmbd daemons \
#              used to provide SMB network services.
```

```
# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

# Check that smb.conf exists.
[ -f /etc/smb.conf ] || exit 0

RETVAL=0

# See how we were called.
case "$1" in
  start)
        echo -n "Starting SMB services: "
        daemon smbd -D
        RETVAL=$?
        echo
        echo -n "Starting NMB services: "
        daemon nmbd -D
        RETVAL2=$?
        echo
        [ $RETVAL -eq 0 -a $RETVAL2 -eq 0 ] && touch /var/lock/subsys/smb || \
          RETVAL=1
        ;;
  stop)
        echo -n "Shutting down SMB services: "
        killproc smbd
        RETVAL=$?
        echo
        echo -n "Shutting down NMB services: "
        killproc nmbd
        RETVAL2=$?
        [ $RETVAL -eq 0 -a $RETVAL2 -eq 0 ] && rm -f /var/lock/subsys/smb
        echo ""
        ;;
  restart)
        $0 stop
        $0 start
        RETVAL=$?
        ;;
  reload)
      echo -n "Reloading smb.conf file: "
        killproc -HUP smbd
        RETVAL=$?
        echo
        ;;
  status)
        status smbd
        status nmbd
        RETVAL=$?
        ;;
  *)
        echo "Usage: $0 {start|stop|restart|status}"
        exit 1
esac

exit $RETVAL
```

Now, make this script executable and change its default permission:
[root@deep]# **chmod 700 /etc/rc.d/init.d/smb**

Configuration of the /etc/pam.d/samba file

Configure your */etc/pam.d/samba* file to use pam authentication.

Create the **samba** file (touch /etc/pam.d/samba) and add:

```
Auth            required      /lib/security/pam_pwdb.so nullok shadow
Account         required      /lib/security/pam_pwdb.so
```

Configuration of the /etc/logrotate.d/samba file

Configure your */etc/logrotate.d/samba* file to rotate each week your log files automatically.

Create the **samba** file (touch /etc/logrotate.d/samba) and add:

```
/var/log/samba/log.nmb {
   notifempty
   missingok
   postrotate
      /usr/bin/killall -HUP nmbd
   endrotate
}

/var/log/samba/log.smb {
   notifempty
   missingok
   postrotate
      /usr/bin/killall -HUP smbd
   endrotate
}
```

Further documentation

For more details, there are several man pages you can read:

```
$ man Samba (7)          - A Windows SMB/CIFS fileserver for UNIX
$ man smb.conf (5)       - The configuration file for the Samba suite
$ man smbclient (1)      - ftp-like client to access SMB/CIFS resources on servers
$ man smbd (8)           - server to provide SMB/CIFS services to clients
$ man smbmnt (8)         - mount smb file system
$ man smbmount (8)       - mount smb file system
$ man smbpasswd (5)      - The Samba encrypted password file
$ man smbpasswd (8)      - change a users SMB password
$ man smbrun (1)         - interface program between smbd and external programs
$ man smbsh (1)          - Allows access to Windows NT filesystem using UNIX commands
$ man smbstatus (1)      - report on current Samba connections
$ man smbtar (1)         - shell script for backing up SMB shares directly to UNIX tape drives
$ man smbumount (8)      - umount for normal users
$ man testparm (1)       - check an smb.conf configuration file for internal correctness
$ man testprns (1)       - check printer name for validity with smbd
```

Securing Samba

Create an encrypted password file

Important: to create a Samba account you must first have a valid Linux account for them. Generate the smbpasswd file from your */etc/passwd* file using the following command:

[root@deep]# **cat /etc/passwd | mksmbpasswd.sh > /etc/smbpasswd**

Create the user account:

[root@deep]# **smbpasswd -a username**
[root@deep]# **chmod 600 /etc/smbpasswd**
[root@deep]# **testparm** (this will verify the smb.conf file for error).

NOTE: See ENCRYPTION.txt in samba/doc/texts/ for more information.

Samba Administrative Tools

smbstatus

smbstatus is a very simple program to list the current Samba connections.

[root@deep]# **smbstatus**

```
Samba version 2.0.5a
Service      uid      gid      pid     machine
----------------------------------------------
tmp          webmaster webmaster  3995   gate     (192.168.1.3) Sat Sep 25 19:40:54 1999

No locked files

Share mode memory usage (bytes):
  1048464(99%) free + 56(0%) used + 56(0%) overhead = 1048576(100%) total
```

Samba Users Tools

smbclient

smbclient is a client that can talk to an SMB/CIFS server. If offers an interface similar to that of the FTP program. Operations include things like getting files from the server to the local machine, putting files from the local machine to the server, retrieving directory information from the server and so on.

[root@deep]# **smbclient *//sbmserver/sharename -U username***
[root@deep]# smbclient //gate/tmp -U webmaster

-Where *//sbmserver* is the name of the server you want to connect to. */sharename* is the directory on this server you want to connect to and, *-U* is your username on this machine.

```
Password:
Domain=[OPENARCH] OS=[Windows NT 4.0] Server=[NT LAN Manager 4.0]
Smb: \>
```

Installed files

```
/usr/sbin/nmbd
/usr/sbin/smbd
/usr/sbin/swat
/usr/sbin/mksmbpasswd.sh
/usr/bin/rpcclient
/usr/bin/smbstatus
/usr/bin/testprns
/usr/man/man1/smbsh.1
/usr/man/man1/smbclient.1
/usr/man/man1/smbrun.1
/usr/man/man1/smbstatus.1
/usr/man/man1/smbtar.1
/usr/man/man1/testparm.1
/usr/man/man1/testprns.1
```

/usr/bin/addtosmbpass
/usr/bin/convert_smbpasswd
/usr/bin/make_smbcodepage
/usr/bin/nmblookup
/usr/bin/smbclient
/usr/bin/smbpasswd
/usr/bin/smbtar
/usr/bin/testparm
/usr/bin/make_printerdef
/usr/bin/mksmbpasswd.sh
/etc/smbpasswd
/etc/smb.conf
/etc/lmhosts
/etc/printers.def
/etc/codepages/
/etc/logrotate.d/samba
/etc/pam.d/samba
/etc/rc.d/init.d/ smb (chkconfig --del smb)
/usr/man/man1/make_smbcodepage.1
/usr/man/man1/nmblookup.1
/usr/man/man7/samba.7
/usr/share/swat/
/var/lock/samba/
/var/log/samba/
/var/spool/samba/
/usr/man/man5/smb.conf.5
/usr/man/man5/lmhosts.5
/usr/man/man5/smbpasswd.5
/usr/man/man8/smbd.8
/usr/man/man8/nmbd.8
/usr/man/man8/swat.8
/usr/man/man8/smbpasswd.8
/usr/man/man8/smbmount.8
/usr/man/man8/smbumount.8
/usr/man/man8/smbmnt.8

Linux OpenLDAP Server

Overview

LDAP (Lightweight Directory Access Protocol) is an open-standard protocol for accessing information services. The protocol runs over Internet transport protocols, such as TCP, and can be used to access stand-alone directory servers or X.500 directories.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
OpenLDAP version number is 1.2.7

Packages

OpenLDAP Homepage: http://www.openldap.org/
You must be sure to download: openldap-version.tgz

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install OpenLDAP, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > fileldap1.txt**’ before and ‘**find /* > fileldap2.txt**’ after you install the software, and use ‘**diff fileldap1.txt fileldap2.txt > diffldap.txt**’ to get a list of what changed.

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# cp openldap-version.tgz /var/tmp
[root@deep]# cd /var/tmp/
[root@deep]# tar xzpf openldap-version.tgz
```

Compile and Optimize

Cd into the new OpenLDAP directory and type the following commands on your terminal:

Edit the **string.h** file (vi +52 include/ac/string.h) and remove the lines:

```
#else
        /* some systems have strdup(), but fail to declare it */
        extern char *(strdup)();
```

Edit the **configure** file (vi +7111 configure) and change the line:

```
SLAPD_LIBS="$SLAPD_LIBS -lwrap"
To:
SLAPD_LIBS="$SLAPD_LIBS -lwrap -lnsl"

CC="egcs" \
CFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--libexecdir=/usr/sbin \
--localstatedir=/var/run \
--sysconfdir=/etc \
--enable-passwd \
--enable-shell \
--enable-wrappers \
--enable-shared

[root@deep]# make depend
[root@deep]# make
[root@deep]# cd tests/
[root@deep]# make
[root@deep]# cd ..
[root@deep]# make install

[root@deep]# install -d -m 700 /var/ldap
[root@deep]# echo localhost > /etc/openldap/ldapserver
[root@deep]# mv -f /usr/sbin/xrpcomp /usr/bin/
[root@deep]# chmod +x /usr/lib/liblber.so.1.0.0
[root@deep]# chmod +x /usr/lib/libldap.so.1.0.0
[root@deep]# strip /usr/lib/liblber.so.1.0.0
[root@deep]# strip /usr/lib/libldap.so.1.0.0
[root@deep]# strip /usr/lib/libldap.a
[root@deep]# strip /usr/lib/liblber.a
[root@deep]# strip /usr/sbin/in.xfingerd
[root@deep]# strip /usr/sbin/go500
[root@deep]# strip /usr/sbin/go500gw
[root@deep]# strip /usr/sbin/mail500
[root@deep]# strip /usr/sbin/rp500
[root@deep]# strip /usr/sbin/fax500
[root@deep]# strip /usr/sbin/rcpt500
[root@deep]# strip /usr/sbin/slapd
[root@deep]# strip /usr/sbin/ldif2ldbm
[root@deep]# strip /usr/sbin/ldif2index
[root@deep]# strip /usr/sbin/ldif2id2entry
[root@deep]# strip /usr/sbin/ldif2id2children
[root@deep]# strip /usr/sbin/ldbmcat
[root@deep]# strip /usr/sbin/ldif
[root@deep]# strip /usr/sbin/centipede
[root@deep]# strip /usr/sbin/ldbmtest
```

```
[root@deep]# strip /usr/sbin/slurpd
[root@deep]# strip /usr/bin/ud
[root@deep]# strip /usr/bin/ldapadd
[root@deep]# strip /usr/bin/ldapsearch
[root@deep]# strip /usr/bin/ldapmodify
[root@deep]# strip /usr/bin/ldapmodrdn
[root@deep]# strip /usr/bin/ldappasswd
[root@deep]# strip /usr/bin/ldapdelete
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf ldap openldap-version.tgz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **slapd.conf** file to */etc/openldap/* directory.
Copy **ldap** script to */etc/rc.d/init.d/* directory.
Make the symbolic rc.d links for **ldap** with the command **chkconfig --add ldap**.
Start your ldap Server with the following command:
[root@deep]# **/etc/rc.d/init.d/ldap start**

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/ldap/slapd.conf file

The */etc/opendap/slapd.conf* file is the main config file for configuring ldap server, permission, password, database type, database location and so on.

Edit the **slap.conf** file (vi /etc/openldap/slapd.conf) and add:

```
#
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#
include         /etc/openldap/slapd.at.conf
include         /etc/openldap/slapd.oc.conf
schemacheck     off
#referral        ldap://ldap.itd.umich.edu

pidfile         /var/run/slapd.pid
argsfile        /var/run/slapd.args

########################################################################
# ldbm database definitions
########################################################################
```

```
database        ldbm
suffix          "o=openarch, c=com"
directory       /var/ldap
rootdn          "cn=admin, o=openarch, c=com"
rootpw          secret
# cleartext passwords, especially for the rootdn, should
# be avoid.  See slapd.conf(5) for details.

# ldbm indexed attribute definitions
index cn,sn,uid
index objectclass pres,eq
index default none
# ldbm access control definitions
defaultaccess read
access to attr=userpassword
        by self write
        by dn="cn=admin, o=openarch, c=com" write
        by * compare
```

Configuration of the /etc/rc.d/init.d/ldap script file

Configure your */etc/rc.d/init.d/ldap* script file to start and stop Ldap Server.

Create the **ldap** script file (touch /etc/rc.d/init.d/ldap) and add:

```
#!/bin/sh
#
# ldap    This shell script takes care of starting and stopping
#         ldap servers (slapd and slurpd).
#
# chkconfig: - 70 40
# description: LDAP stands for Lightweight Directory Access Protocol, used \
#          for implementing the industry standard directory services.
# processname: slapd
# config: /etc/openldap/slapd.conf
# pidfile: /var/run/slapd.pid

# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

[ -f /usr/sbin/slapd ] || exit 0
[ -f /usr/sbin/slurpd ] || exit 0

RETVAL=0

# See how we were called.
case "$1" in
  start)
      # Start daemons.
      echo -n "Starting ldap: "
      daemon slapd
          RETVAL=$?
          if [ $RETVAL -eq 0 ]; then
        if grep -q "^replogfile" /etc/openldap/slapd.conf; then
                  daemon slurpd
```

```
                RETVAL=$?
                [ $RETVAL -eq 0 ] && pidof slurpd | cut -f 1 -d " " > /var/run/slurpd
        fi
          fi
    echo
          [ $RETVAL -eq 0 ] && touch /var/lock/subsys/ldap
    ;;
  stop)
    # Stop daemons.
          echo -n "Shutting down ldap: "
          killproc slapd
          RETVAL=$?
          if [ $RETVAL -eq 0 ]; then
            if grep -q "^replogfile" /etc/openldap/slapd.conf; then
                killproc slurpd
                RETVAL=$?
            fi
          fi
          echo
    if [ $RETVAL -eq 0 ]; then
            rm -f /var/lock/subsys/ldap
            rm -f /var/run/slapd.args
          fi
    ;;
  status)
          status slapd
          RETVAL=$?
          if [ $RETVAL -eq 0 ]; then
            if grep -q "^replogfile" /etc/openldap/slapd.conf; then
                status slurpd
                RETVAL=$?
            fi
          fi
          ;;
  restart)
          $0 stop
          $0 start
          RETVAL=$?
          ;;
  reload)
          killproc -HUP slapd
          RETVAL=$?
          if [ $RETVAL -eq 0 ]; then
            if grep -q "^replogfile" /etc/openldap/slapd.conf; then
                killproc -HUP slurpd
                RETVAL=$?
            fi
          fi
          ;;
  *)
          echo "Usage: $0 start|stop|restart|status}"
          exit 1
esac

exit $RETVAL
```

Now, make this script executable and change its default permission:
[root@deep]# **chmod 700 /etc/rc.d/init.d/ldap**

Further documentation

For more details, there are several man pages you can read:

$ man ldapd (8)	- LDAP X.500 Protocol Daemon
$ man ldapdelete (1)	- ldap delete entry tool
$ man ldapfilter.conf (5)	- configuration file for LDAP get filter routines
$ man ldapfriendly (5)	- data file for LDAP friendly routines
$ man ldapmodify, ldapadd (1)	- ldap modify entry and ldap add entry tools
$ man ldapmodrdn (1)	- ldap modify entry RDN tool
$ man ldappasswd (1)	- change the password of an LDAP entry
$ man ldapsearch (1)	- ldap search tool
$ man ldapsearchprefs.conf (5)	- configuration file for LDAP search preference routines
$ man ldaptemplates.conf (5)	- configuration file for LDAP display template routines
$ man ldif (5)	- LDAP Data Interchange Format
$ man slapd (8)	- Stand-alone LDAP Daemon
$ man slapd.conf (5)	- configuration file for slapd, the stand-alone LDAP daemon
$ man slurpd (8)	- Standalone LDAP Update Replication Daemon
$ man ud (1)	- interactive LDAP Directory Server query program

OpenLDAP Creation and Maintenance Tools

Creating a database off-line

This method is best if you have many thousands of entries to create, which would take an unacceptably long time using the ldapadd method. This tool read the slapd configuration file and an input file containing a text representation of the entries to add.

[root@deep]# **ldif2ldbm -i *<inputfile>* -f *<slapdconfigfile>***
[root@deep]# ldif2ldbm -i my-database-files -f /etc/ldap/slapd.conf

-where <*inputfile*> specifies the LDIF input file containing the entries to add in text form. <*slapdconfigfile*> specifies the slapd configuration file that tells where to create the indexes, what indexes to create, etc.

Creating a database over LDAP

With this method you use the ldapadd tool to add entries, just like you would once the database is created. You should be sure to set the following options in your **slapd.conf** file before starting *slapd*.

suffix *<dn>*

This option says what entries are to be held by this database. You should set this to the DN of the root of the subtree you are trying to create.

For example:
suffix “o=openarch, c=com”

You should be sure to specify a directory where the index files should be created

directory *<directory>*

For example:
directory /var/ldap

You need to make it so you can connect to slapd as somebody with permission to add entries. This is done through the following two options in the database definition:

rootdn *<dn>*

rootpw *<passwd>* /* Remember to use crypto password here !!! */

These options specify a DN and password that can be used to authenticate as the "superuser" entry of the database (i.e., the entry allowed to do anything). The DN and password specified here will always work, regardless of whether the entry named actually exists or has the password given.

Finally, you should make sure that the database definition contains the index definitions you want:

index {<attrlist> | default} [pres,eq,approx,sub,none]

For example, to index the cn, sn, uid and objectclass attributes the following index configuration lines could be used.

index cn,sn,uid

index objectclass pres,eq

index default none

Once you have configured things to your liking, start up slapd, connect with your LDAP client, and start adding entries. For example, to add a “Europe Mourani” entry using the ldapadd tool, you could create a file called */tmp/newentry* with the contents:

cn=Europe Mourani, o=openarch, c=com
cn=Europe Mourani
description= Marketing relation - emourani@openarch.com
objectClass=top
objectClass=person

-And then use a command like this to actually create the entry:

[root@deep]# **ldapadd -f /tmp/newentry -D "cn=admin, o=openarch, c=com" -W**

The above command assumes that you have set rootdn to "cn=admin, o=openarch, c=com" and rootpw to "secret". You will be prompted to enter the password:

[root@deep]# ldapadd -f /tmp/newentry -D "cn=admin, o=openarch, c=com" -W
Enter LDAP Password :

ldapmodify

The ldapmodify command opens a connection to an LDAP server, binds, and modifies or adds entries. The entry information is read from standard input or from file through the use of the -f option.

Input file format for the ldapmodify command.

Assuming that the file /tmp/entry exists and has the contents:
cn=Europe Mourani, o=openarch, c=com
- mail=emourani@old.com # will delete the old mail address for Europe Mourani in the database.
+mail=emourani@new.com # will add the new mail address for Europe Mourani in the database.

The command
[root@deep]# **ladpmodify -D ‘cn=Admin, o=openarch, c=com’ -W -f <*inputfile*>**

-Will replace the contents of the “Europe Mourani” entry’s mail attribute with the value emourani@new.com.

OpenLDAP Users Tools

Search on LDAP for entries

ldapsearch opens a connection to an LDAP server, binds, and performs a search using the filter.

[root@deep]# **ldapsearch -b <*dn*> <*attrs*>**
[root@deep]# **ldapsearch -b 'o=openarch.com' 'cn=a*'**

- Will retrieve all entries and values beginning by the letter **a** and will printed to standard output.

Installed files

> /etc/openldap/*
> /usr/bin/ldapsearch
> /usr/bin/ldapmodify
> /usr/bin/ldapdelete
> /usr/bin/ldapmodrdn
> /usr/bin/ldappasswd
> /usr/bin/ldapadd
> /usr/bin/ud
> /usr/include/ldap.h
> /usr/include/lber.h
> /usr/include/ldap_cdefs.h
> /usr/include/disptmpl.h
> /usr/include/srchpref.h
> /usr/lib/liblber.so.1.0.0
> /usr/lib/liblber.so.1
> /usr/lib/liblber.so
> /usr/lib/liblber.la
> /usr/lib/liblber.a
> /usr/lib/libldap.so.1.0.0
> /usr/lib/libldap.so.1
> /usr/lib/libldap.so
> /usr/lib/libldap.la
> /usr/lib/libldap.a
> /usr/man/man1/ud.1
> /usr/man/man1/ldapdelete.1
> /usr/man/man1/ldapmodify.1
> /usr/man/man1/ldapadd.1
> /usr/man/man1/ldapmodrdn.1
> /usr/man/man1/ldappasswd.1
> /usr/man/man1/ldapsearch.1
> /usr/man/man3/cldap_open.3
> /usr/man/man3/cldap_close.3
> /usr/man/man3/cldap_search_s.3
> /usr/man/man3/cldap_setretryinfo.3
> /usr/man/man3/lber-decode.3
> /usr/man/man3/lber-encode.3
> /usr/man/man3/ldap.3
> /usr/man/man3/cldap.3
> /usr/man/man3/ldap_abandon.3
> /usr/man/man3/ldap_add.3
> /usr/man/man3/ldap_add_s.3
> /usr/man/man3/ldap_bind.3
> /usr/man/man3/ldap_bind_s.3
> /usr/man/man3/ldap_simple_bind.3
> /usr/man/man3/ldap_simple_bind_s.3
> /usr/man/man3/ldap_kerberos_bind_s.3
> /usr/man/man3/ldap_kerberos_bind1.3
> /usr/man/man3/ldap_kerberos_bind1_s.3
> /usr/man/man3/ldap_explode_dn.3
> /usr/man/man3/ldap_explode_dns.3
> /usr/man/man3/ldap_dn2ufn.3
> /usr/man/man3/ldap_is_dns_dn.3
> /usr/man/man3/ldap_get_values.3
> /usr/man/man3/ldap_get_values_len.3
> /usr/man/man3/ldap_value_free.3
> /usr/man/man3/ldap_value_free_len.3
> /usr/man/man3/ldap_count_values.3
> /usr/man/man3/ldap_count_values_len.3
> /usr/man/man3/ldap_getfilter.3
> /usr/man/man3/ldap_init_getfilter.3
> /usr/man/man3/ldap_init_getfilter_buf.3
> /usr/man/man3/ldap_getfilter_free.3
> /usr/man/man3/ldap_getfirstfilter.3
> /usr/man/man3/ldap_getnextfilter.3
> /usr/man/man3/ldap_setfilteraffixes.3
> /usr/man/man3/ldap_build_filter.3
> /usr/man/man3/ldap_modify.3
> /usr/man/man3/ldap_modify_s.3
> /usr/man/man3/ldap_mods_free.3
> /usr/man/man3/ldap_modrdn.3
> /usr/man/man3/ldap_modrdn_s.3
> /usr/man/man3/ldap_modrdn2.3
> /usr/man/man3/ldap_modrdn2_s.3
> /usr/man/man3/ldap_open.3
> /usr/man/man3/ldap_init.3
> /usr/man/man3/ldap_result.3
> /usr/man/man3/ldap_msgfree.3
> /usr/man/man3/ldap_search.3
> /usr/man/man3/ldap_search_s.3
> /usr/man/man3/ldap_search_st.3
> /usr/man/man3/ldap_searchprefs.3
> /usr/man/man3/ldap_init_searchprefs.3
> /usr/man/man3/ldap_init_searchprefs_buf.3
> /usr/man/man3/ldap_free_searchprefs.3
> /usr/man/man3/ldap_first_searchobj.3
> /usr/man/man3/ldap_next_searchobj.3
> /usr/man/man3/ldap_sort.3
> /usr/man/man3/ldap_sort_entries.3
> /usr/man/man3/ldap_sort_values.3
> /usr/man/man3/ldap_sort_strcasecmp.3
> /usr/man/man3/ldap_ufn.3
> /usr/man/man3/ldap_ufn_search_s.3
> /usr/man/man3/ldap_ufn_search_c.3
> /usr/man/man3/ldap_ufn_search_ct.3
> /usr/man/man3/ldap_ufn_setprefix.3
> /usr/man/man3/ldap_ufn_setfilter.3

> /usr/man/man3/ldap_kerberos_bind2.3
> /usr/man/man3/ldap_kerberos_bind2_s.3
> /usr/man/man3/ldap_unbind.3
> /usr/man/man3/ldap_unbind_s.3
> /usr/man/man3/ldap_set_rebind_proc.3
> /usr/man/man3/ldap_cache.3
> /usr/man/man3/ldap_enable_cache.3
> /usr/man/man3/ldap_disable_cache.3
> /usr/man/man3/ldap_destroy_cache.3
> /usr/man/man3/ldap_flush_cache.3
> /usr/man/man3/ldap_uncache_entry.3
> /usr/man/man3/ldap_uncache_request.3
> /usr/man/man3/ldap_set_cache_options.3
> /usr/man/man3/ldap_charset.3
> /usr/man/man3/ldap_set_string_translators.3
> /usr/man/man3/ldap_enable_translation.3
> /usr/man/man3/ldap_translate_from_t61.3
> /usr/man/man3/ldap_translate_to_t61.3
> /usr/man/man3/ldap_t61_to_8859.3
> /usr/man/man3/ldap_8859_to_t61.3
> /usr/man/man3/ldap_compare.3
> /usr/man/man3/ldap_compare_s.3
> /usr/man/man3/ldap_delete.3
> /usr/man/man3/ldap_delete_s.3
> /usr/man/man3/ldap_disptmpl.3
> /usr/man/man3/ldap_init_templates.3
> /usr/man/man3/ldap_init_templates_buf.3
> /usr/man/man3/ldap_free_templates.3
> /usr/man/man3/ldap_first_disptmpl.3
> /usr/man/man3/ldap_next_disptmpl.3
> /usr/man/man3/ldap_oc2template.3
> /usr/man/man3/ldap_tmplattrs.3
> /usr/man/man3/ldap_first_tmplrow.3
> /usr/man/man3/ldap_next_tmplrow.3
> /usr/man/man3/ldap_first_tmplcol.3
> /usr/man/man3/ldap_next_tmplcol.3
> /usr/man/man3/ldap_entry2text.3
> /usr/man/man3/ldap_entry2text_search.3
> /usr/man/man3/ldap_vals2text.3
> /usr/man/man3/ldap_entry2html.3
> /usr/man/man3/ldap_entry2html_search.3
> /usr/man/man3/ldap_vals2html.3
> /usr/man/man3/ldap_error.3
> /usr/man/man3/ldap_perror.3
> /usr/man/man3/ld_errno.3
> /usr/man/man3/ldap_result2error.3
> /usr/man/man3/ldap_errlist.3
> /usr/man/man3/ldap_err2string.3
> /usr/man/man3/ldap_first_attribute.3
> /usr/man/man3/ldap_next_attribute.3
> /usr/man/man3/ldap_first_entry.3
> /usr/man/man3/ldap_next_entry.3
> /usr/man/man3/ldap_count_entries.3
> /usr/man/man3/ldap_friendly.3
> /usr/man/man3/ldap_friendly_name.3
> /usr/man/man3/ldap_free_friendlymap.3
> /usr/man/man3/ldap_get_dn.3
> /usr/man/man3/ldap_ufn_timeout.3
> /usr/man/man3/ldap_url.3
> /usr/man/man3/ldap_is_ldap_url.3
> /usr/man/man3/ldap_url_parse.3
> /usr/man/man3/ldap_free_urldesc.3
> /usr/man/man3/ldap_url_search.3
> /usr/man/man3/ldap_url_search_s.3
> /usr/man/man3/ldap_url_search_st.3
> /usr/man/man5/ldif.5
> /usr/man/man5/ldap.conf.5
> /usr/man/man5/ldapfilter.conf.5
> /usr/man/man5/ldapfriendly.5
> /usr/man/man5/ldapsearchprefs.conf.5
> /usr/man/man5/ldaptemplates.conf.5
> /usr/man/man5/slapd.conf.5
> /usr/man/man5/slapd.replog.5
> /usr/man/man5/ud.conf.5
> /usr/man/man8/centipede.8
> /usr/man/man8/chlog2replog.8
> /usr/man/man8/edb2ldif.8
> /usr/man/man8/go500.8
> /usr/man/man8/go500gw.8
> /usr/man/man8/in.xfingerd.8
> /usr/man/man8/ldapd.8
> /usr/man/man8/ldbmcat.8
> /usr/man/man8/ldif.8
> /usr/man/man8/ldif2ldbm.8
> /usr/man/man8/ldif2index.8
> /usr/man/man8/ldif2id2entry.8
> /usr/man/man8/fax500.8
> /usr/man/man8/ldif2id2children.8
> /usr/man/man8/mail500.8
> /usr/man/man8/rcpt500.8
> /usr/man/man8/slapd.8
> /usr/man/man8/slurpd.8
> /usr/sbin/slapd
> /usr/sbin/ldif2ldbm
> /usr/sbin/ldif2index
> /usr/sbin/ldif2id2entry
> /usr/sbin/ldif2id2children
> /usr/sbin/ldbmcat
> /usr/sbin/ldif
> /usr/sbin/centipede
> /usr/sbin/ldbmtest
> /usr/sbin/in.xfingerd
> /usr/sbin/go500
> /usr/sbin/go500gw
> /usr/sbin/mail500
> /usr/sbin/rp500
> /usr/sbin/fax500
> /usr/sbin/xrpcomp
> /usr/sbin/rcpt500
> /usr/sbin/slurpd
> /usr/share/openldap
> /usr/share/openldap/ldapfriendly
> /usr/share/openldap/go500gw.help
> /usr/share/openldap/rcpt500.help

Overview

Postgres, developed originally in the UC Berkeley Computer Science Department, pioneered many of the object-relational concepts now becoming available in some commercial databases. It provides SQL92/SQL3 language support, transaction integrity, and type extensibility. PostgreSQL is a public domain, open source descendant of this original Berkeley code.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
PostgreSQL version number is 6.5.2

Packages

PostgreSQL Homepage: http://www.postgresql.org/

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install it, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filesql1.txt**’ before and ‘**find /* > filesql2.txt**’ after you install the tarball, and use ‘**diff filesql1.txt filesql2.txt > diffsql.txt**’ to get a list of what changed.

Compilation

Create the Postgres Superuser Account (**postgres** is commonly used).

```
[root@deep]# useradd -M -o -r -d /var/lib/pgsql -s /bin/bash -c "PostgreSQL Server" postgres >/dev/null
2>&1 || :
```

Decompress the tarball (tar.gz).

```
[root@deep]# cp postgresql-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf postgresql-version.tar.gz
```

Compile and Optimize

Cd into the new PosgreSQL directory and type the following on your terminal:

```
[root@deep]# cd src
CC=”egcs” \
./configure \
--prefix=/usr \
--enable-hba \
--enable-locale \
--with-perl \
--with-odbc
```

[root@deep]# vi +208 Makefile.global and:

Change: CFLAGS= -I$(SRCDIR)/include -I$(SRCDIR)/backend

To:
CFLAGS= -I$(SRCDIR)/include -I$(SRCDIR)/backend **-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions**

```
[root@deep]# make all
[root@deep]# cd ..
[root@deep]# make -C src install
[root@deep]# make -C src/man install
[root@deep]# make -C src/interfaces/perl5 install
[root@deep]# mkdir -p /usr/include/pgsql
[root@deep]# mv /usr/include/access  /usr/include/pgsql/
[root@deep]# mv /usr/include/commands  /usr/include/pgsql/
[root@deep]# mv /usr/include/executor  /usr/include/pgsql/
[root@deep]# mv /usr/include/lib  /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq  /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq++  /usr/include/pgsql/
[root@deep]# mv /usr/include/port  /usr/include/pgsql/
[root@deep]# mv /usr/include/utils  /usr/include/pgsql/
[root@deep]# mv /usr/include/fmgr.h /usr/include/pgsql/
[root@deep]# mv /usr/include/os.h /usr/include/pgsql/
[root@deep]# mv /usr/include/config.h /usr/include/pgsql/
[root@deep]# mv /usr/include/c.h /usr/include/pgsql/
[root@deep]# mv /usr/include/postgres.h /usr/include/pgsql/
[root@deep]# mv /usr/include/postgres_ext.h /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq-fe.h /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq-int.h /usr/include/pgsql/
[root@deep]# mv /usr/include/ecpgerrno.h /usr/include/pgsql/
[root@deep]# mv /usr/include/ecpglib.h /usr/include/pgsql/
[root@deep]# mv /usr/include/ecpgtype.h /usr/include/pgsql/
[root@deep]# mv /usr/include/sqlca.h /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq++.H /usr/include/pgsql/
[root@deep]# mkdir -p /usr/lib/pgsql
[root@deep]# mv /usr/lib/*source /usr/lib/pgsql/
[root@deep]# mv /usr/lib/*sample /usr/lib/pgsql/
[root@deep]# cd src/interfaces/perl5/
[root@deep]# mkdir -p /usr/lib/pgsql/perl5
[root@deep]# cp Pg.pm /usr/lib/pgsql/perl5/
[root@deep]# cd src/interfaces/perl5/blib/arch/auto/Pg/
[root@deep]# cp Pg.so Pg.bs /usr/lib/pgsql/perl5/
[root@deep]# mkdir -p /var/lib/pgsql
[root@deep]# chown -R postgres.postgres /var/lib/pgsql/
[root@deep]# chmod 755 /usr/lib/libpsqlodbc.so*
[root@deep]# chmod 755 /usr/lib/libpq.so*
[root@deep]# chmod 755 /usr/lib/libecpg.so*
[root@deep]# chmod 755 /usr/lib/libpq++.so*
[root@deep]# strip /usr/bin/postgres
[root@deep]# strip /usr/bin/postmaster
[root@deep]# strip /usr/bin/ecpg
[root@deep]# strip /usr/bin/pg_id
[root@deep]# strip /usr/bin/pg_version
[root@deep]# strip /usr/bin/pg_dump
[root@deep]# strip /usr/bin/pg_passwd
[root@deep]# strip /usr/bin/psql
[root@deep]# rm -f /usr/lib/global1.description
[root@deep]# rm -f /usr/lib/local1_template1.description
[root@deep]# rm -f /usr/odbcinst.ini
[root@deep]# su postgres
[postgres@deep]# initdb –pglib=/usr/lib/pgsql –pgdata=/var/lib/pgsql
[postgres@deep]# chmod 640 /var/lib/pgsql/pg_pwd
[postgres@deep]# exit
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf postgresql-version/ postgresql-version.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **postgresql** script to */etc/rc.d/init.d/* directory.

```
[root@deep]# cd /etc/rc.d/rc3.d/
[root@deep]# ln -s ../init.d/postgresql S99postgresql
[root@deep]# cd /etc/rc.d/rc5.d/
[root@deep]# ln -s ../init.d/postgresql S99postgresql
```

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/rc.d/init.d/postgresql script file

Configure your */etc/rc.d/init.d/postgres* script file to start and stop Postgresql Server.

Create the **postgresql** script file (touch /etc/rc.d/init.d/postgresql) and add:

```
#! /bin/sh
# postgresql    This is the init script for starting up the PostgreSQL
#               server
#
# chkconfig: 345 85 15
# processname: postmaster
# pidfile: /var/run/postmaster.pid
#

# Source function library.
. /etc/rc.d/init.d/functions

# Get config.
. /etc/sysconfig/network

# Check that networking is up.
# Pretty much need it for postmaster.
[ ${NETWORKING} = "no" ] && exit 0

[ -f /usr/bin/postmaster ] || exit 0

# This script is slightly unusual in that the name of the daemon (postmaster)
# is not the same as the name of the subsystem (postgresql)

# See how we were called.
case "$1" in
  start)
        echo -n "Starting postgresql service: "
```

```
        su -l postgres -c '/usr/bin/postmaster -S -D/var/lib/pgsql'
        sleep 1
        pid=`pidof postmaster`
        echo -n "postmaster [$pid]"
        touch /var/lock/subsys/postgresql
        echo $pid > /var/run/postmaster.pid
        echo
        ;;
  stop)
        echo -n "Stopping postgresql service: "
        killproc postmaster
        sleep 2
        rm -f /var/run/postmaster.pid
        rm -f /var/lock/subsys/postgresql
        echo
        ;;
  status)
        status postmaster
        ;;
  restart)
        $0 stop
        $0 start
        ;;
  *)
        echo "Usage: postgresql {start|stop|status|restart}"
        exit 1
esac

exit 0
```

Now, make this script executable and change its default permission:
[root@deep]# **chmod 700 /etc/rc.d/init.d/postgresql**

Commands

The commands listed bellow are some that we use often in our regular use but much more exist and you must check the man page for more details and information.

Client connections can be restricted by IP address and/or user name via the **pg_hba.conf** file in PG_DATA.

To define a new user, run the **createuser** utility program.
[root@deep]# su postgres
[postgres@deep]# createuser

To remove a user, run the **destroyuser** utility program.
[root@deep]# su postgres
[postgres@deep]# destroyuser

To create database
[root@deep]# su createduser (see above).
[createduser@deep]# createdb dbname (the name of the database).

or with the Postgres terminal monitor program (psql)
[root@deep]# psql template1
template1➔ create database foo;

To destroy database name (the name of the database).

Other useful Postgres terminal monitor program (psql).

Connect to the new database:
template1➔ \c foo

Create a table
foo➔ create table bar (I int4, c char(16));

Inspect the new table
foo➔ \d bar

Drop a table, index, view:
foo➔ drop table table_name;
foo➔ drop index index_name;
foo➔ drop view view_name;

Insert into: (once a table is created, it can be filled using the command…)
foo➔ insert into table_name (name_of_attr1, name_of_attr2, name_of_attr3)
foo➔ values (value1, value2, value3);

Installed files

/etc/rc.d/init.d/postgresql
/etc/rc.d/rc3.d/S99postgresql
/etc/rc.d/rc5.d/S99postgresql
/usr/bin/postgres
/usr/bin/postmaster
/usr/bin/ecpg
/usr/bin/pg_id
/usr/bin/pg_version
/usr/bin/pg_dump
/usr/bin/pg_dumpall
/usr/bin/pg_upgrade
/usr/bin/pg_passwd
/usr/bin/psql
/usr/bin/cleardbdir
/usr/bin/createdb
/usr/bin/createlang
/usr/bin/createuser
/usr/bin/destroydb
/usr/bin/destroylang
/usr/bin/destroyuser
/usr/bin/initdb
/usr/bin/vacummdb
/usr/bin/initlocation
/usr/bin/ipcclean
/usr/include/iodbc/
/usr/include/pgsql/
/usr/lib/perl5/5.00503/i386-linux/perllocal.pod
/usr/lib/perl5/man/man3/Pg.3
/usr/lib/perl5/site_perl/5.005/i386-linux/auto/
/usr/lib/perl5/site_perl/5.005/i386-linux/Pg.pm
/usr/lib/libpq.a
/usr/lib/libpq.so.2.0
/usr/lib/libpq.so.2
/usr/lib/libpq.so
/usr/lib/libecpg.a
/usr/lib/libecpg.so.3.0.0
/usr/lib/libecpg.so.3
/usr/lib/libecpg.so
/usr/lib/libpq++.a
/usr/lib/libpq++.so.3.0
/usr/lib/libpq++.so.3
/usr/lib/libpq++.so
/usr/lib/libpsqlodbc.a
/usr/lib/libpsqlodbc.so.0.25
/usr/lib/libpsqlodbc.so.0
/usr/lib/libpsqlodbc.so
/usr/lib/plpgsql.so
/usr/lib/pgsql/
/usr/man/man1/cleardbdir.1
/usr/man/man1/createdb.1
/usr/man/man1/createuser.1
/usr/man/man1/destroydb.1
/usr/man/man1/destroyuser.1
/usr/man/man1/ecpg.1
/usr/man/man1/initdb.1
/usr/man/man1/initlocation.1
/usr/man/man1/ipcclean.1
/usr/man/man1/pg_dump.1
/usr/man/man1/pg_dumpall.1
/usr/man/man1/pg_passwd.1
/usr/man/man1/pg_upgrade.1
/usr/man/man1/postgres.1
/usr/man/man1/postmaster.1
/usr/man/man1/psql.1
/usr/man/man3/libpq.3
/usr/man/man3/catalogs.3
/usr/man/man5/pg_hba.conf.5
/usr/man/manl/
/var/lib/pgsql/

Overview

Squid is the result of efforts by numerous individuals from the Internet community. Development is led by Duane Wessels of the National Laboratory for Applied Network Research and funded by the National Science Foundation. Squid is derived from the ``cached'' software from the ARPA-funded Harvest research project. In our compilation and configuration we'll configure Squid to run as an httpd-accelerator to get more performance (really more). An accelerator caches incoming requests for outgoing data (i.e., that which you publish to the world). It takes load away from your HTTP server and internal network. You move the server away from port 80 (or whatever your published port is), and substitute the accelerator, which then pulls the HTTP data from the ``real'' HTTP server (only the accelerator needs to know where the real server is). The outside world sees no difference (apart from an increase in speed, with luck).

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account "root".
Squid version number is 2.2.STABLE5

Packages

Squid Homepage: http://squid.nlanr.net/
You must be sure to download: squid-2.2.STABLE5-src.tar.gz

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install Squid, and one afterwards, and then compare them using '**diff**' to find out what file it placed where. Simply run '**find /* > filesquid1.txt**' before and '**find /* > filesquid2.txt**' after you install the software, and use '**diff filesquid1.txt filesquid2.txt > diffsquid.txt**' to get a list of what changed.

Compilation

Squid Proxy Server can't run as superuser root, for this reason we'll create a special user with less privilege for running Squid Proxy Server.

Add user squid to passwd file:

```
[root@deep]# /usr/sbin/useradd -d /cache/ -r -s /dev/null squid >/dev/null 2>&1
[root@deep]# mkdir /cache/ (we don't need to make this command because we are already create this directory when we are partitioning our disk. If this partition doesn't exist so execute this command to create the directory).
[root@deep]# chown -R squid.squid /cache/
```

Decompress the tarball (tar.gz).

```
[root@deep]# cp squid-version.STABLEz-src.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf squid-version.STABLEz-src.tar.gz
```

Configure and Optimize

Cd into the new Squid directory and type the following commands on your terminal:

[root@deep]# vi +18 icons/Makefile.in and change:

DEFAULT_ICON_DIR = $(**libexecdir**)/icons

[root@deep]# vi +34 src/Makefile.in and change:

DEFAULT_CACHE_LOG = $(localstatedir)**/log/squid/**cache.log
DEFAULT_ACCESS_LOG = $(localstatedir)**/log/squid/**access.log
DEFAULT_STORE_LOG = $(localstatedir)**/log/squid/**store.log
DEFAULT_PID_FILE = $(localstatedir)**/run/**squid.pid
DEFAULT_SWAP_DIR = **/cache**
DEFAULT_ICON_DIR = **$(libexecdir)**/icons

malloc

Many users have found improved performance when linking Squid with an external malloc library, such as GNU malloc. To make Squid use GNU malloc follows these simple steps:

1. Download the GNU malloc source, available from one of The GNU FTP Mirror sites (http://www.gnu.org/order/ftp.html).

2. Compile GNU malloc
   [root@deep]# tar xzpf malloc.tar.gz
   [root@deep]# cd malloc
   [root@deep]# vi Makefile and uncomment the line: CPPFLAGS = -DUSG # in the Makefile
   [root@deep]# export CC=egcs
   [root@deep]# make

3. Copy libmalloc.a to your system's library directory and be sure to name it *libgnumalloc.a*.
   [root@deep]# **cp libmalloc.a /usr/lib/libgnumalloc.a**

4. (Optional) Copy the GNU malloc.h to your system's include directory and be sure to name it *gnumalloc.h*. This step is not required, but if you do this, then Squid will be able to use the *mstat()* function to report memory usage statistics on the cachemgr info page.
   [root@deep]# **cp malloc.h /usr/include/gnumalloc.h**

Compile and Optimize

Return into the new Squid directory and type the following commands on your terminal:

[root@deep]# **export CACHE_HTTP_PORT=80**

```
CC="egcs" \
CFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--exec-prefix=/usr \
--bindir=/usr/sbin \
--libexecdir=/usr/lib/squid \
--localstatedir=/var \
--sysconfdir=/etc/squid \
--enable-cache-digests \
--enable-poll
```

```
[root@deep]# make -f makefile
[root@deep]# make install
```

```
[root@deep]# mkdir -p /var/log/squid
[root@deep]# rm -rf /var/logs/
[root@deep]# chown squid.squid /var/log/squid/
[root@deep]# chmod 750 /var/log/squid/
[root@deep]# chmod 750 /cache/
[root@deep]# rm -f /usr/sbin/RunCache
[root@deep]# rm -f /usr/sbin/RunAccel
[root@deep]# strip /usr/sbin/squid
[root@deep]# strip /usr/sbin/client
[root@deep]# strip /usr/lib/squid/dnsserver
[root@deep]# strip /usr/lib/squid/unlinkd
[root@deep]# strip /usr/lib/squid/cachemgr.cgi
[root@deep]# mv /usr/lib/squid/cachemgr.cgi /home/httpd/cgi-bin/
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf squid-version/ squid-version.STABLEz.tar.gz
[root@deep]# rm -rf malloc/ malloc.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **squid.conf** file to */etc/squid/* directory.
Copy **squid** script to */etc/rc.d/init.d/* directory.
Copy **squid** file to */etc/logrotate.d/* directory.
Make the symbolic rc.d links for **squid** with the command **chkconfig –add squid**.
Start your new proxy Server with the following command:
[root@deep]# **/etc/rc.d/init.d/squid start**

By default squid script will not start automatically the proxy server on Red Hat Linux when you reboot the server. You can change it default by executing the following command:

```
[root@deep]# setup
```

Choose "system services", select "squid" and place an "*" on it, click "ok" and "quit".

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/squid/squid.conf file

Configure your */etc/squid/squid.conf* file to be in httpd-accelerator mode. You must put your httpd (Apache) daemon running on port 81. With Apache you can do it like this in *httpd.conf*:

```
Port 81
```

Then, in your *squid.conf* file, you must specify the hostname address as the accelerator:

```
httpd_accel_host 192.168.1.1
```

```
httpd_accel_port 81
```

Edit the **squid.conf** file (vi /etc/squid/squid.conf) and add:

```
http_port 80
icp_port 0
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
cache_dir /cache 200 16 256
half_closed_clients off
memory_pools off
cache_mem 16 MB
acl all src 0.0.0.0/0.0.0.0
http_access allow all
cache_mgr admin
cache_effective_user squid
cache_effective_group squid
httpd_accel_host 192.168.1.1
httpd_accel_port 81
log_icp_queries off
store_avg_object_size 20 KB
store_objects_per_bucket 20
buffered_logs on
```

Configuration of the /etc/rc.d/init.d/squid script file

Configure your */etc/rc.d/init.d/squid* script file to start and stop Squid Internet Object Cache. This script have been modified to setup swap cache for Squid in */cache* instate of */var/spool/squid*.

Create the **squid** script file (touch /etc/rc.d/init.d/squid) and add:

```
#!/bin/bash
# squid             This shell script takes care of starting and stopping
#                   Squid Internet Object Cache
#
# chkconfig: - 90 25
# description: Squid - Internet Object Cache. Internet object caching is \
#          a way to store requested Internet objects (i.e., data available \
#          via the HTTP, FTP, and gopher protocols) on a system closer to the \
#          requesting site than to the source. Web browsers can then use the \
#          local Squid cache as a proxy HTTP server, reducing access time as \
#          well as bandwidth consumption.
# pidfile: /var/run/squid.pid
# config: /etc/squid/squid.conf

PATH=/usr/bin:/sbin:/bin:/usr/sbin
export PATH

# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

# check if the squid conf file is present
[ -f /etc/squid/squid.conf ] || exit 0

# determine the name of the squid binary
```

```
[ -f /usr/sbin/squid ] && SQUID=squid
[ -z "$SQUID" ] && exit 0

# determine which one is the cache_swap directory
CACHE_SWAP=`sed -e 's/#.*//g' /etc/squid/squid.conf | \
        grep cache_dir | sed -e 's/cache_dir//' | \
        cut -d ' ' -f 2`
[ -z "$CACHE_SWAP" ] && CACHE_SWAP=/cache

# default squid options
# -D disables initial dns checks. If you most likely will not to have an
#    internet connection when you start squid, uncomment this
SQUID_OPTS="-D"

RETVAL=0
case "$1" in
start)
   echo -n "Starting $SQUID: "
   for adir in $CACHE_SWAP; do
      if [ ! -d $adir/00 ]; then
            echo -n "init_cache_dir $adir... "
            $SQUID -z -F 2>/dev/null
        fi
   done
   $SQUID $SQUID_OPTS &
   RETVAL=$?
   echo $SQUID
   [ $RETVAL -eq 0 ] && touch /var/lock/subsys/$SQUID
   ;;

stop)
   echo -n "Stopping $SQUID: "
   $SQUID -k shutdown &
   RETVAL=$?
   if [ $RETVAL -eq 0 ] ; then
        rm -f /var/lock/subsys/$SQUID
        while : ; do
                [ -f /var/run/squid.pid ] || break
                sleep 2 && echo -n "."
        done
        echo "done"
   else
        echo
   fi
   ;;

reload)
   $SQUID $SQUID_OPTS -k reconfigure
   exit $?
   ;;

restart)
   $0 stop
   $0 start
   ;;

status)
   status $SQUID
   $SQUID -k check
   exit $?
   ;;
```

```
probe)
   exit 0;
   ;;

*)
   echo "Usage: $0 {start|stop|status|reload|restart}"
   exit 1
esac

exit $RETVAL
```

Now, make this script executable and change its default permission:
[root@deep]# **chmod 700 /etc/rc.d/init.d/squid**

Configuration of the /etc/logrotate.d/squid file

Configure your */etc/logrotate.d/squid* file to rotate each week your log files automatically.

Create the **squid** file (touch /etc/logrotate.d/squid) and add:

```
/var/log/squid/access.log {
   weekly
   rotate 5
   copytruncate
   compress
   notifempty
   missingok
}
/var/log/squid/cache.log {
   weekly
   rotate 5
   copytruncate
   compress
   notifempty
   missingok
}

/var/log/squid/store.log {
   weekly
   rotate 5
   copytruncate
   compress
   notifempty
   missingok
# This script asks squid to rotate its logs on its own.
# Restarting squid is a long process and it is not worth
# doing it just to rotate logs
   postrotate
     /usr/sbin/squid -k rotate
   endscript
}
```

Securing Squid

More control on mounting a file system

You can have more control on mounting a file system like */cache* with some nifty options like noexec, nodev, and nosuid. This can be setup in */etc/fstab*:

Edit your **fstab** file (vi /etc/fstab) and add:

/dev/sda8 /cache ext2 nosuid,nodev,noexec 1 2

-Which mean for <*nodev*> do not interpret character or block special devices on the file system, for <*nosuid*> do not allow set-user-identifier or set-group-identifier bits to take effect and for <*noexec*> do not allow execution of any binaries on the mounted file system.

Optimizing Squid

Increases the system limit on open files

Increases the current process limit. I verified that a process on Red Hat 6.0 (2.2.5) could open at least 31000 file descriptors this way. Another fellow has verified that a process on 2.2.12 can open at least 90000 file descriptors this way (with appropriate limits). The upper bound seems to be available memory.

Edit the **profile** file (vi /etc/profile) and add the following line:
ulimit -n 90000

NOTE: In older 2.2 kernels, though, the number of open files per process is still limited to 1024, even with the above changes.

The ulimit

Note: Linux itself has a "Max Processes" per user limit. Add this to your root "**.bashrc"** file or whatever script your particular shell uses (in our case we'll use the /etc/profile file to make it on in all our system, process and users):

Edit the **profile** file (vi /etc/profile) and add the following line:
ulimit -u unlimited

You must exit and re-login before starting your new Squid! Otherwise you will run into problems. To verify that you are ready to go, make sure that when you type **ulimit -a** as root, it shows "unlimited" next to **max user processes**

The atime

For a busy cache, you may consider turning off access-time updates for the cache file system (this is a common trick for high-volume NNTP servers). This prevents the OS updating the "last accessed" time on files that it reads, reducing the number of disk writes.

To set the attribute, use the following command:

[root@deep]# **chattr -R +A /cache/**

The noatime attribute

Linux has a mount option for filesystems call **noatime**. This option can be added to the mount options field in */etc/fstab*. When a filesystem is mounted with this option, read accesses to the file will no longer result in an update to the atime information associated with the file. The atime info is generally not all that useful, so the lacks of updates to this field are not often relevant. The importance of the **noatime** setting is that it eliminates the need to make writes to the filesystem for files, which are simply being read. Since writes tend to be somewhat expensive, this can result in measurable performance gains. Note that the wtime information will continue to be updated anytime the file is written to.

[root@deep]# vi /etc/fstab and add:

E.I: /dev/sda8 /cache ext2 nosuid,nodev,noexec,**noatime** 1 2

Reboot your system and then test your results with the command:
[root@deep]# **cat /proc/mounts**

The bdflush parameter
This documentation is for the sysctl files in */proc/sys/vm* and is valid for Linux kernel version 2.2. The files in this directory can be used to tune the operation of the virtual memory (VM) subsystem of the Linux kernel, and one of the files (bdflush) also has a little influence on disk usage. This file (bdflush) controls the operation of the bdflush kernel daemon. We generally use this command to improve filesystem performance:

echo "100 1200 128 512 15 5000 500 1884 2">/proc/sys/vm/bdflush

Add the above commands to */etc/rc.d/rc.local* and you'll not have to type it again the next time you reboot your system.

Be changing some values from the default and the system seems more responsive, ie: it waits a little more to write to disk and thus avoids some disk access contention.

Look at */usr/src/linux/Documentation/sysctl/vm.txt* for more information on how to improve kernel parameters related to virtual memory, disk cache, swap, etc.

The ip_local_port_range parameter
This documentation is for the sysctl files in */proc/sys/net/ipv4/ip_local_port_range* and is valid for Linux kernel version 2.2. ip_local_port_range - 2 INTEGERS.

Defines the local port range that is used by TCP and UDP to choose the local port. The first number is the first (port), the second the last local port number. For high-usage systems change this to 32768-61000.

echo "32768 61000" > /proc/sys/net/ipv4/ip_local_port_range

Add the above commands to ***/etc/rc.d/rc.local*** file and you'll not have to type it again the next time if you reboot your system.

Physical memory
The most important resource for Squid is physical memory. Your processor does not need to be ultra-fast. Your disk system will be the major bottleneck, so fast disks are important for high-volume caches. Do not use IDE disks if you can help it.

Installed files

/etc/logrotate.d/squid
/etc/rc.d/init.d/squid (chkconfig –del squid)
/etc/squid/
/usr/lib/squid/
/usr/sbin/squid
/usr/sbin/client
/usr/sbin/RunCache
/usr/sbin/RunAccel
/var/log/squid/

Linux Apache Server

Overview

Because Apache is a complex package there are a lot of installation variants and options. For this different documents exists which explain special things: For more information's read this document when you want to install Apache under Unix. Here, I explain how to compile and optimize Apache with a lot options like mod_ssl, mod_perl, mod_php4, LDAP… because I don't want to make several individual document's from each one. Fill free to compile what you want, E.I.: For Apache + PHP4 but without mod_ssl or PostgreSQL or… follow the section that speak about Apache and PHP4 and skip the rest.

As you noticed above there are a lot of possibilities, variants and options for installing Apache. So, in the following we provide some step-by-step examples where you can see how to build Apache with other third-party modules. For simplification we assume some prerequisites for each example. If these don't fit your situation you have to adjust the steps. This session is geared for new Apache Webmasters, and describes how to install an Apache server and get it running. It also covers some basic ways in which you can adjust the configuration to improve the server's performance. In our configuration and installation we'll run Apache as non root-user and in a chrooted environment for optimal security.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account "root".
Apache version number is 1.3.9
Mod_Perl version number is 1.21
Mod_SSL version number is 2.4.6-1.3.9
PHP version number is 3.0.12
MM version number is 1.0.12

Packages

Apache Homepage: http://www.apache.org/
Mod_ssl Homepage: http://www.modssl.org/
OpenSSL Homepage: http://www.openssl.org/
MM Homepage: http://www.engelschall.com/sw/mm/
Imap Homepage: http://www.washington.edu/imap/
PostgreSQL Homepage: http://www.postgresql.org/index.html
OpenLDAP Homepage: http://www.openldap.org/
Mod_perl Homepage: http://perl.apache.org/
Php Homepage: http://www.php.net/

Prerequisites

OpenSSL should be already installed on your system (E.I.: if you want Apache+mod_ssl)
PosgreSQL should be already installed on your system (E.I.: if you want Apache+PHP3+Pgsql)
Mm should be already installed on your system (E.I.: if you want Apache+mod_ssl+mm)
OpenLDAP should be already installed on your system (E.I.: is you want Apache+PHP3+LDAP)
Perl should be already installed on your system (E.I.: if you want Apache+mod_perl)

Imap should be already installed on your system (E.I.: if you want Apache+PHP3+imap)

Why do I need to use MM?
Build the MM Shared Memory library when you want shared memory support in Apache/EAPI. For instance this allows mod_ssl to use a high-performance RAM-based session cache instead of a disk-based one.

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install Apache, and one afterwards, and then compare them using '**diff**' to find out what file it placed where. Simply run '**find /* > fileapache1.txt**' before and '**find /* > fileapache2.txt**' after you install the software's, and use '**diff fileapache1.txt fileapache2.txt > diffapache.txt**' to get a list of what changed.

Compilation

Decompress the tarballs (tar.gz).

```
[root@deep]# cp apache_version.tar.gz /var/tmp
[root@deep]# cp mod_ssl-version-version.tar.gz /var/tmp
[root@deep]# cp mod_perl-version.tar.gz /var/tmp
[root@deep]# cp php-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp/
[root@deep]# tar xzpf apache_version.tar.gz
[root@deep]# tar xzpf mod_ssl-version-version.tar.gz
[root@deep]# tar xzpf mod_perl-version.tar.gz
[root@deep]# tar xzpf php-version.tar.gz
```

Compile and Optimize

Apply mod-ssl to Apache source tree.
Cd into the new mod_ssl directory and type the following commands on your terminal:

```
CC="egcs" \
CFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions" \
./configure \
--with-apache=../apache_1.3.9 \
--with-crt=/etc/ssl/certs/server.crt \
--with-key=/etc/ssl/private/server.key
```

Cd into the new Apache directory (*cd ../apache-version*) and type the following commands on your terminal:

[root@deep]# vi +331 src/include/httpd.h and change:

```
#define HARD_SERVER_LIMIT 256
To
#define HARD_SERVER_LIMIT 1024
```

Pre-configure Apache for PHP3's configure step.
Cd into the new Apache directory (*cd ../apache-version*) and type the following commands on your terminal:

```
CC="egcs" \
OPTIM="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions" \
CFLAGS="-DDYNAMIC_MODULE_LIMIT=0" \
./configure \
```

```
--prefix=/home/httpd \
--bindir=/usr/bin \
--sbindir=/usr/sbin \
--libexecdir=/usr/lib/apache \
--includedir=/usr/include/apache \
--sysconfdir=/etc/httpd/conf \
--localstatedir=/var \
--runtimedir=/var/run \
--logfiledir=/var/log/httpd \
--datadir=/home/httpd \
--proxycachedir=/var/cache/httpd \
--mandir=/usr/man
```

Configure PHP3 and apply it to the Apache source tree.
Cd into the new php3 directory (*cd ../php-version*) and type the following commands on your terminal:

```
[root@deep]# vi +46 ext/pgsql/php3_pqsql.h

#include </usr/include/pgsql/libpq-fe.h>
#include </usr/include/pgsql/libpq/libpq-fs.h>


CC="egcs" \
CFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions -
I/usr/include/openssl" \
./configure \
--prefix=/usr \
--with-config-file-path=/etc/httpd \
--with-imap \
--enable-safe-mode \
--with-exec-dir=/usr/bin \
--disable-debug \
--with-apache=../apache_1.3.9 \
--with-pgsql \ (if you want database PostgreSQL)
--with-ldap \ (if you want LDAP database light directory)
--enable-memory-limit=yes \
--enable-debug=no

[root@deep]# make
[root@deep]# make install
```

Apply mod_perl to Apache source tree and build/install the Perl-side of mod_perl.
Cd into the new mod_perl directory (*cd../mod_perl-version*) and type the following commands on your terminal:

```
perl Makefile.PL \
EVERYTHING=1 \
APACHE_SRC=../apache_1.3.9/src \
USE_APACI=1 \
PREP_HTTPD=1 \
DO_HTTPD=1

[root@deep]# make
[root@deep]# make install
```

Build/Install Apache with mod_ssl + mm + mod_perl and PHP3.
Cd into the new Apache directory (*cd ../apache-version*) and type the following commands on your terminal:

```
SSL_BASE=SYSTEM \ (require for mod_ssl).
EAPI_MM=SYSTEM \ (require for mm).
CC="egcs" \
OPTIM="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions" \
CFLAGS="-DDYNAMIC_MODULE_LIMIT=0" \
./configure \
--prefix=/home/httpd \
--bindir=/usr/bin \
--sbindir=/usr/sbin \
--libexecdir=/usr/lib/apache \
--includedir=/usr/include/apache \
--sysconfdir=/etc/httpd/conf \
--localstatedir=/var \
--runtimedir=/var/run \
--logfiledir=/var/log/httpd \
--datadir=/home/httpd \
--proxycachedir=/var/cache/httpd \
--mandir=/usr/man \
--add-module=src/modules/experimental/mod_mmap_static.c \ (if you are intended to use mod_mmap).
--add-module=src/modules/standard/mod_auth_db.c \ (if you are intended to use mod_auth).
--enable-module=ssl \ (require for mod_ssl).
--enable-rule=SSL_SDBM \ (require for mod_ssl).
--disable-rule=SSL_COMPAT \ (require for mod_ssl).
--activate-module=src/modules/php3/libphp3.a \ (require for php).
--enable-module=php3 \ (require for php).
--activate-module=src/modules/perl/libperl.a \ (require for mod_perl).
--enable-module=perl \ (require for mod_perl).
--disable-module=include \
--disable-module=status \
--disable-module=userdir \
--disable-module=negotiation \
--disable-module=autoindex \
--disable-module=asis \
--disable-module=imap \
--disable-module=env \
--disable-module=actions

[root@deep]# make
[root@deep]# make install

[root@deep]# rm -f /usr/sbin/apachectl
[root@deep]# rm -f /usr/man/man8/apachectl.8
[root@deep]# rm -rf /home/httpd/icons/
[root@deep]# cd /var/tmp/php-version
[root@deep]# install -m 644 php3.ini.dist /etc/httpd/php.ini
[root@deep]# rm -rf /etc/httpd/conf/ssl.crl/
[root@deep]# rm -rf /etc/httpd/conf/ssl.crt/
[root@deep]# rm -rf /etc/httpd/conf/ssl.csr/
[root@deep]# rm -rf /etc/httpd/conf/ssl.key/
[root@deep]# rm -rf /etc/httpd/conf/ssl.prm/
[root@deep]# rm -f /etc/httpd/conf/srm.conf srm.conf.default access.conf access.conf.default
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf apache-version/ apache-version.tar.gz mod_ssl-version-version/ mod_ssl-version-version.tar.gz php-version/ php-version.tar.gz mod_perl-version/ mod_perl-version.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and

other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **httpd.conf** file to */etc/httpd/conf/* directory.
Copy **apache** file to */etc/logrotate.d/* directory.
Copy **httpd** script to */etc/rc.d/init.d/* directory.
Make the symbolic rc.d links for **httpd** with the command **chkconfig –add httpd**.
Start your new Apache server with the following command:
[root@deep]# **/etc/rc.d/init.d/httpd start**

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/httpd/conf/httpd.conf file

Configure your */etc/httpd/conf/httpd.conf* file. The following example is a minimal configuration for Apache with SSL protocol.

Edit the **httpd.conf** file (vi /etc/httpd/conf/httpd.conf) and add:

```
ServerType standalone
ResourceConfig /dev/null
AccessConfig /dev/null
ServerRoot "/etc/httpd"
PidFile /var/run/httpd.pid
Timeout 300
KeepAlive On
MaxKeepAliveRequests 0
KeepAliveTimeout 15
MinSpareServers 16
MaxSpareServers 64
StartServers 16
MaxClients 512
MaxRequestsPerChild 100000
Port 80

<IfDefine SSL>
Listen 192.168.1.1:80
Listen 192.168.1.1:443
</IfDefine>

User www
Group www
ServerAdmin admin@openarch.com
ServerName www.openarch.com
DocumentRoot "/home/httpd/ona"
Include conf/mmap.conf

<Directory />
   Options None
   AllowOverride None
   Order deny,allow
   Deny from all
```

```
</Directory>

<Directory /home/httpd>
   Options None
   AllowOverride None
   Order allow,deny
   Allow from all
</Directory>

<Files .pl>
   Options None
   AllowOverride None
   Order deny,allow
   Deny from all
</Files>

DirectoryIndex index.htm index.php index.html default.html index.cgi
UseCanonicalName On
TypesConfig /etc/httpd/conf/mime.types
DefaultType text/plain
HostnameLookups Off

ErrorLog /var/log/httpd/error_log
LogLevel warn
LogFormat "%h %v %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
/etc/httpd/conf/httpd.conf
SetEnvIf Request_URI \.gif$ gif-image
CustomLog /var/log/httpd/access_log combined env=!gif-image
ServerSignature Off

ScriptAlias /cgi-bin/  "/home/httpd/cgi-bin/"
 <Directory "/home/httpd/cgi-bin">
   AllowOverride None
   Options None
   Order allow,deny
   Allow from all
 </Directory>

AddType application/x-httpd-php3 .php3
AddType application/x-httpd-php3-source .phps

ErrorDocument 500 "The server made a boo boo.
ErrorDocument 404 http://www.openarch.com/error.htm
ErrorDocument 403 "Access Forbidden -- Go away.
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0

<IfDefine SSL>
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
</IfDefine>

<IfModule mod_ssl.c>
SSLPassPhraseDialog  builtin
SSLSessionCache        dbm:/var/run/ssl_scache
SSLSessionCacheTimeout  300
SSLMutex  file:/var/run/ssl_mutex
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
```

```
SSLLog      /var/log/httpd/ssl_engine_log
SSLLogLevel warn
</IfModule>

<IfDefine SSL>
<VirtualHost _default_:443>
DocumentRoot "/home/httpd/ona"
ServerName www.openarch.com
ServerAdmin admin@openarch.com
ErrorLog /var/log/httpd/error_log
SSLEngine on
SSLCipherSuite ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile    /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
SSLCACertificatePath    /etc/ssl/certs
SSLCACertificateFile    /etc/ssl/certs/ca.crt
SSLCARevocationPath     /etc/ssl/crl
SSLVerifyClient none
SSLVerifyDepth  10

SSLOptions +ExportCertData +StrictRequire
SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown
SetEnvIf Request_URI \.gif$ gif-image
CustomLog /var/log/httpd/ssl_request_log \
        "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b" env=!gif-image
</VirtualHost>
</IfDefine>
```

NOTE: if you use the mod_php3 module with your Apache server don't forget to include in your */etc/httpd/conf/httpd.conf* file the following line:

```
AddType application/x-httpd-php3 .php3
AddType application/x-httpd-php3-source .phps
```

NOTE: if you use the mod_perl module with your Apache server don't forget to include in your */etc/httpd/conf/httpd.conf* file the following line to be able to see status of your different perls modules on the server:

```
<Location /perl-status>
        SetHandler  perl-script
        PerlHandler Apache::Status
        Order deny,allow
        Deny from all
        Allow from 192.168.1.3
 </Location>
```

Configuration of the /etc/logrotate.d/apache file

Configure your */etc/logrotate.d/apache* file to rotate each week your log files automatically.

Create the **apache** file (touch /etc/logrotate.d/apache) and add:

```
/var/log/httpd/access_log {
   missingok
   postrotate
      /usr/bin/killall -HUP httpd
   endscript
}

/var/log/httpd/error_log {
```

```
    missingok
    postrotate
        /usr/bin/killall -HUP httpd
    endscript
}

/var/log/httpd/ssl_request_log {
    missingok
    postrotate
        /usr/bin/killall -HUP httpd
    endscript
}
```

Configuration of the /etc/rc.d/init.d/httpd script file

Configure your */etc/rc.d/init.d/httpd* script file to start and stop Apache server.

Create the **httpd** script file (touch /etc/rc.d/init.d/httpd) and add:

```
#!/bin/sh
#
# Startup script for the Apache Web Server
#
# chkconfig: 345 85 15
# description: Apache is a World Wide Web server.  It is used to serve \
#              HTML files and CGI.
# processname: httpd
# pidfile: /var/run/httpd.pid
# config: /etc/httpd/conf/httpd.conf


# Source function library.
. /etc/rc.d/init.d/functions

# See how we were called.
case "$1" in
  start)
        echo -n "Starting httpd: "
        daemon httpd -DSSL
        echo
        touch /var/lock/subsys/httpd
        ;;
  stop)
        echo -n "Shutting down http: "
        killproc httpd
        echo
        rm -f /var/lock/subsys/httpd
        rm -f /var/run/httpd.pid
        ;;
  status)
        status httpd
        ;;
  restart)
        $0 stop
        $0 start
        ;;
  reload)
        echo -n "Reloading httpd: "
        killproc httpd -HUP
        echo
        ;;
```

```
 *)
        echo "Usage: $0 {start|stop|restart|reload|status}"
        exit 1
esac

exit 0
```

Now, make this script executable and change its default permission:
[root@deep]# **chmod 700 /etc/rc.d/init.d/httpd**

NOTE: The “-DSSL” option will start Apache on mode SSL. If you wan to start Apache on regular mode remove the “-DSSL” option near “daemon httpd”.

Securing Apache

There are several important configuration options that affect security. The first is the user that the web server runs as (User and Group, in httpd.conf). It is best to choose a user that has as few privileges possible - ideally, a user created just for the purpose of running the webserver daemon. Create this user (We generally call it www) and ensure that it has minimal access to the system and its functions.

```
[root@deep]# groupadd -g 80 www
[root@deep]# useradd -g 80 -u 80 www
```

Change some important permission files and directory of your WebServer.

```
[root@deep]# chmod 511 /usr/sbin/httpd
[root@deep]# chmod 750 /etc/httpd/conf/
[root@deep]# chmod 750 /var/log/httpd/
```

You can create an http subdirectory which is modifiable by other users -- since root never executes any files out of there, and shouldn't be creating files in there (**ex: */home/httpd/ona***).

By default, Apache usually comes with automatic indexing of directories enabled (IndexOptions in httpd.conf). This means any requests for a directory that don't find a index file, will build an index of what is in the directory. In many cases, this is a security issue, as you many only want people seeing files that you specifically link to. To turn this off, you need to remove read permissions from the directory (but not the files inside) - that is, **chmod 311**. Depending on the ownership of the directory, it should look like:

```
[root@deep]# cd /home/httpd/
[root@deep]# chmod 311 ona
[root@deep]# ls -la

d-wx--x--x   13 webmaster webmaster     1024 Jul 28 08:12 ona
```

Then, any requests for this protected directory should return an error message something like:

```
Forbidden
You don't have permission to access /protected/ on this server.
```

More control on mounting a file system

You can have more control on mounting a file system like */chroot* with some nifty options like nosuid (for chroot file system refer to section “Running Apache in a chroot jail” bellow). This can be setup in */etc/fstab*:

Edit your /etc/fstab file (vi /etc/fstab) and add depending of your needs:

/dev/sda7 /chroot ext2 nosuid 1 2

-Which mean for <*nosuid*> do not allow set-user-identifier or set-group-identifier bits to take.

Create the .dbmpasswd password file for authentication
Needed only if you thing that you'll use an access file authentication for your site.

```
[root@deep]# chmod 750 /usr/bin/dbmmanage
[root@deep]# /usr/bin/dbmmanage  /etc/httpd/.dbmpasswd adduser username
```

Running Apache in a chroot jail
This part focuses on preventing Apache from being used as a point of break-in to the system hosting it. Apache by default run **as a non-root user**, which will limit any damage to what can be done as a normal user with a local shell. Of course, allowing what amounts to an anonymous guest account falls rather short of the security requirements for most Apache servers, so an additional step can be taken - that is, **running Apache in a chroot jail**.

The main benefit of a chroot jail is that the jail will limit the portion of the filesystem the daemon can see to the root directory of the jail. Additionally, since the jail only needs to support Apache, the programs available in the jail can be extremely limited. Most importantly, there is no need for setuid-root programs, which, given the right (or wrong...) bug, can be used to gain root access and break out of the jail.

Chrooting apache is no easy task and has a tendency to break things. Before we embark on this, we need to first decide whether it is beneficial for you to do so. Some pros and cons are (but most certainly not limited to):

Pros:

- If apache is ever compromised, the attacker will not have access to the entire file system. Also make sure apache is running as its own unique user/group, and not something that's overly used, such as "nobody." Consider the scenario where a web server is running as nobody (or any other overly used UID/GID) and compromised. The cracker can now access any other processes running as nobody from within the chroot.

- Poorly written CGI scripts that may, for example, allow someone to email your */etc/passwd* file will not work.

Cons:

- The extra libraries you'll need to have in the chroot. Using hardlinks or compiling your binaries statically will help here. Static binaries are also nifty because they eliminate the possibility -- however infinitesimal it may be -- of someone replacing your libc or other shared library with some hostile wrapper lib and having apache run it unknowingly.

- Generally speaking, the more nifty stuff your web server does, the more difficult it will be to not break that functionality. For example, if you use any Perl/CGI, you will need to copy the needed binaries and perl libraries to the appropriate spot within the chroot space. The same applies for SSL, PHP and other.

NOTE: The chrooted configuration listed bellow suppose that you're compiled your Apache server with **mod_ssl**, **mod_php** and **mod_auth_db** for password authentication. The differences of what you're compiled with your Apache web server resides in which libraries and binaries we'll need to copy in our chrooted lib directory. Remember that if you've compiled Apache to use **mod_perl** you must copy the needed binary and perl libraries to the chrooted directory. Perl

reside in */usr/lib/perl5* directory and in the case you use perl, copy the perl directories to */chroot/httpd/usr/lib/perl5/*. Don't forget to create this directory (/chroot/httpd/usr/lib/perl5) in your chrooted structure before coping.

Find the shared library dependencies of httpd. These will need to be copied into the chroot jail later.

[root@deep]# **ldd /usr/sbin/httpd**

libpam.so.0 => /lib/libpam.so.0 (0x40016000)
libm.so.6 => /lib/libm.so.6 (0x4001f000)
libdl.so.2 => /lib/libdl.so.2 (0x4003b000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x4003e000)
libnsl.so.1 => /lib/libnsl.so.1 (0x4006b000)
libresolv.so.2 => /lib/libresolv.so.2 (0x40081000)
libdb.so.3 => /lib/libdb.so.3 (0x40090000)
libc.so.6 => /lib/libc.so.6 (0x400cb000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)

Make a note of the files listed above; you will need these later.

Step 1:
Add a new user id and a new group id if this is not already done for running httpd. This is important because running it as root defeats the purpose of the jail, and using a different user id that already exists on the system can allow your services to access each others' resources. Think multi-layer security.

These are sample user and group id numbers. Check */etc/passwd* and */etc/group* for a free uid/gid number. We'll use 80.

[root@deep]# **groupadd -g 80 www**
[root@deep]# **useradd -g 80 -u 80 www**

Step 2:
Set up the chroot environment. First we need to create the chrooted Apache structure. We use */chroot/httpd* for the chrooted Apache. The */chroot/httpd* is just a directory on a different partition where we've decided to put apache for more paranoid security.

[root@deep]# **/etc/rc.d/init.d/httpd stop** ← if Apache is already installed and run on your system.
[root@deep]# **mkdir /chroot/httpd**

Next, create the rest of directories like the following:

[root@deep]# **mkdir /chroot/httpd/dev**
[root@deep]# **mkdir /chroot/httpd/lib**
[root@deep]# **mkdir /chroot/httpd/etc**
[root@deep]# **mkdir -p /chroot/httpd/usr/sbin**
[root@deep]# **mkdir -p /chroot/httpd/var/run**
[root@deep]# **mkdir -p /chroot/httpd/var/log/httpd**
[root@deep]# **chmod 750 /chroot/httpd/var/log/httpd/**
[root@deep]# **mkdir -p /chroot/httpd/home/httpd**

Copy the main configuration directory, the configuration files, the cgi-bin directory, the root directory and the httpd program:

[root@deep]# **cp -r /etc/httpd /chroot/httpd/etc/**
[root@deep]# **cp -r /home/httpd/cgi-bin /chroot/httpd/home/httpd/**
[root@deep]# **cp -r /home/httpd/your-DocumentRoot /chroot/httpd/home/httpd/**

[root@deep]# **mknod /chroot/httpd/dev/null c 1 3**
[root@deep]# **chmod 666 /chroot/httpd/dev/null**
[root@deep]# **cp /usr/sbin/httpd /chroot/httpd/usr/sbin/**
[root@deep]# **cp -r /etc/ssl /chroot/httpd/etc/** ← require only if you use mod_ssl.
[root@deep]# **chmod 600 /chroot/httpd/etc/ssl/certs/ca.crt**
[root@deep]# **chmod 600 /chroot/httpd//etc/ssl/certs/server.crt**
[root@deep]# **chmod 600 /chroot/httpd/etc/ssl/private/ca.key**
[root@deep]# **chmod 600 /chroot/httpd/etc/ssl/private/server.key**

We need the */chroot/httpd/etc, /chroot/httpd/dev, /chroot/httpd/lib, /chroot/httpd/usr/sbin, /chroot/httpd/var/run, /chroot/httpd/home/httpd* and */chroot/httpd/var/log/httpd* directories because, from the point of the chroot, we're sitting at /.

Since we are compiled apache to use shared libraries; we need to install them into the chroot directory structure. Use **ldd /chroot/httpd/usr/sbin/httpd** to find out which libraries are needed. The output (depending of what you've compiled with Apache) will be something similar to:

libpam.so.0 => /lib/libpam.so.0 (0x40016000)
libm.so.6 => /lib/libm.so.6 (0x4001f000)
libdl.so.2 => /lib/libdl.so.2 (0x4003b000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x4003e000)
libnsl.so.1 => /lib/libnsl.so.1 (0x4006b000)
libresolv.so.2 => /lib/libresolv.so.2 (0x40081000)
libdb.so.3 => /lib/libdb.so.3 (0x40090000)
libc.so.6 => /lib/libc.so.6 (0x400cb000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)

Copy the shared libraries identified above:

[root@deep]# **cp /lib/libpam.so.0 /chroot/httpd/lib/**
[root@deep]# **cp /lib/libm.so.6 /chroot/httpd/lib/**
[root@deep]# **cp /lib/libdl.so.2 /chroot/httpd/lib/**
[root@deep]# **cp /lib/libcrypt.so.1 /chroot/httpd/lib/**
[root@deep]# **cp /lib/libnsl* /chroot/httpd/lib/**
[root@deep]# **cp /lib/libresolv* /chroot/httpd/lib/**
[root@deep]# **cp /lib/libdb.so.3 /chroot/httpd/lib/**
[root@deep]# **cp /lib/libc.so.6 /chroot/httpd/lib/**
[root@deep]# **cp /lib/ld-linux.so.2 /chroot/httpd/lib/**

You'll also need the following extra libraries for some network functions like resolving:

[root@deep]# **cp /lib/libnss_compat* /chroot/httpd/lib/**
[root@deep]# **cp /lib/libnss_dns* /chroot/httpd/lib/**
[root@deep]# **cp /lib/libnss_files* /chroot/httpd/lib/**

We now need to copy passwd and group files inside the */chroot/httpd/etc* chrooted directory. The concept here is how ftpd uses passwd and group files. Lastly, run **ldconfig -r /chroot/httpd/** to create a cache for any libraries that may be in the chroot.

[root@deep]# **cp /etc/passwd /chroot/httpd/etc/**
[root@deep]# **cp /etc/group /chroot/httpd/etc/**
[root@deep]# **ldconfig -r /chroot/httpd/**

NOTE: The command **ldconfig** must be executed after httpd process have been started and not before.

Next, remove all entries except for the user that apache runs as in both files (passwd and group). You will also need **/etc/resolv.conf, /etc/nsswitch.conf** and **/etc/hosts** files.

Edit the **passwd** file (vi /chroot/httpd/etc/passwd) and delete all entries except the user apache run as (in our configuration is "www"):

Edit the **group** file (vi /chroot/httpd/etc/group) and delete all entries except the group apache run as (in our configuration is "www"):

[root@deep]# **cp /etc/resolv.conf /chroot/httpd/etc/**
[root@deep]# **cp /etc/hosts /chroot/httpd/etc/**
[root@deep]# **cp /etc/nsswitch.conf /chroot/httpd/etc/**

Copy the localtime file to the jail so that log entries are adjusted for your local timezone properly:

[root@deep]# **cp /etc/localtime /chroot/httpd/etc/**

Remove unnecessary old files and directories:

[root@deep]# **rm -rf /var/log/httpd/**
[root@deep]# **rm -rf /etc/httpd/**
[root@deep]# **rm -rf /home/httpd/**
[root@deep]# **rm -f /usr/sbin/httpd**

Step 3:
Tell syslogd about the new chrooted service.

Normally, processes talk to syslogd through */dev/log*. As a result of the chroot jail, this won't be possible, so syslogd needs to be told to listen to */chroot/httpd/dev/log*. To do this, edit the syslog startup script to specify additional places to listen.

Edit the **syslog** script (vi /etc/rc.d/init.d/syslog) to change the line:

daemon syslogd -m 0
To read:
daemon syslogd -m 0 -a /chroot/httpd/dev/log

Step 4:
Edit the **httpd** script (vi /etc/rc.d/init.d/httpd) to change the line:

daemon httpd
To read:
/usr/sbin/chroot /chroot/httpd/ /usr/sbin/httpd -DSSL

rm -f /var/run/httpd.pid
To read:
rm -f /chroot/httpd/var/run/httpd.pid

Red Hat's init scripts' daemon() function doesn't allow alternate PID files to be specified, but that won't affect the operation of the start and stop actions of the httpd init scripts since they will be called from outside the chroot jail.

Step 5:
Test the new chrooted configuration! Restart syslogd:

root@deep]# **/etc/rc.d/init.d/syslog stop**
root@deep]# **/etc/rc.d/init.d/syslog start**

Now, start the new chrooted Apache:

Whew, we're finished! Try it out. **/etc/rc.d/init.d/httpd start**. If you don't get any errors, do a **ps auwx | grep httpd** and see if we're running. If so, lets check to make sure it's chrooted by picking out one of the process numbers of httpd and doing **ls -la /proc/that_process_number/root/**. If you see:
dev
etc
home
lib
usr
var
congratulations!

As mentioned above, if you use Perl, you'll need to copy or hardlinks any system libraries, perl libraries (*/usr/lib/perl5*), and binaries into the chroot area. The same applies for SSL, PHP and other.

Configuration of the new /etc/logrotate.d/apache file

Now Apache logs file reside on */chroot/var/log/httpd* directory instead of */var/log/httpd*, for this reason we need to modify our */etc/logrotate.d/httpd* file to point to the new chrooted directory. Also we're compiled Apache with mod_ssl and we'll add one more line to permit logrotate program to rotate the ssl_request_log file. Configure your */etc/logrotate.d/apache* file to rotate each week your log files automatically.

Create the **apache** file (touch /etc/logrotate.d/apache) and add:

```
/var/log/httpd/access_log {
   missingok
   postrotate
      /usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
   endscript
}

/var/log/httpd/error_log {
   missingok
   postrotate
      /usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
   endscript
}

/var/log/httpd/ssl_request_log {
   missingok
   postrotate
      /usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
   endscript
}
```

Optimizing Apache

The static file

For static file, compile mod_mmap_static (if you are follow what I described in the compilation time above, this is already done --add-module-../mod_mmap_static.c) into Apache (see http://www.apache.org/docs/mod/mod_mmap_static.html) and configure Apache to memory-map the static documents, e.g. by creating a config file like this as root:

```
[root@deep]# find /home/httpd/htdocs -type f -print | sed -e 's/.*/mmapfile &/' > /etc/httpd/conf/mmap.conf
```

And including mmap.conf in your Apache config file like this:

[root@deep]# vi /etc/httpd/conf/httpd.conf and add: **Include conf/mmap.conf** somewhere in the file.

The ulimit
NOTE: Linux itself has a "Max Processes" per user limit. Add this to your root "**.bashrc"** file or whatever script your particular shell uses (in our case we'll use the /etc/profile file to make it on in all our system, process and users):

Edit the **profile** file (vi /etc/profile) and add the following line:
ulimit -u unlimited

You must exit and re-login before starting your new Apache! Otherwise you will run into problems. To verify that you are ready to go, make sure that when you type **ulimit -a** as user www, it shows "unlimited" next to **max user processes**.

```
[root@deep]# su www
[www@deep]# ulimit -a

core file size (blocks)     0
data seg size (kbytes)      unlimited
file size (blocks)          unlimited
max memory size (kbytes)    unlimited
stack size (kbytes)         8192
cpu time (seconds)          unlimited
max user processes          unlimited  ← this line.
pipe size (512 bytes)       8
open files                  1024
virtual memory (kbytes)     2105343
```

Note: you may also do ulimit -u unlimited at the command prompt before starting httpd instead of adding it to the "**.bashrc"** file, but I always forgot, so I just added it in the "**.bashrc"** file as a safety net. Another good place to put ulimit -u unlimited is in the httpd startup file in **/etc/rc.d/init.d/httpd**.

Increases the system limit on open files
Increases the current process' limit. I verified that a process on Red Hat 6.0 (2.2.5) could open at least 31000 file descriptors this way. Another fellow has verified that a process on 2.2.12 can open at least 90000 file descriptors this way (with appropriate limits). The upper bound seems to be available memory.

Edit the **profile** file (vi /etc/profile) and add the following line:
ulimit -n 90000

You must exit and re-login. To verify that you are ready to go, make sure that when you type **ulimit -a** as user www, it shows "90000" next to **open files**.

```
[root@deep]# su www
[www@deep]# ulimit -a

core file size (blocks)     0
data seg size (kbytes)      unlimited
file size (blocks)          unlimited
max memory size (kbytes)    unlimited
stack size (kbytes)         8192
cpu time (seconds)          unlimited
max user processes          unlimited
```

```
pipe size (512 bytes)     8
open files                90000 ← this line.
virtual memory (kbytes)   2105343
```

In older 2.2 kernels, though, the number of open files per process is still limited to 1024, even with the above changes.

The noatime

Linux has a mount option for filesystems call **noatime**. This option can be added to the mount options field in */etc/fstab*. When a filesystem is mounted with this option, read accesses to the file will no longer result in an update to the atime information associated with the file. The atime info is generally not all that useful, so the lacks of updates to this field are not often relevant. The importance of the **noatime** setting is that it eliminates the need to make writes to the filesystem for files, which are simply being read. Since writes tend to be somewhat expensive, this can result in measurable performance gains. Note that the wtime information will continue to be updated anytime the file is written to.

Edit the **fstab** file (vi /etc/fstab) and add noatime to:

```
/dev/sda7       /chroot       ext2   nosuid,nodev,noatime       1  2
```

Reboot your system and then test your results with the command:

```
[root@deep]# cat /proc/mounts
```

The ip_local_port_range parameter

This documentation is for the sysctl files in */proc/sys/net/ipv4/ip_local_port_range* and is valid for Linux kernel version 2.2. ip_local_port_range - 2 INTEGERS.

Defines the local port range that is used by TCP and UDP to choose the local port. The first number is the first (port), the second the last local port number. For high-usage systems change this to 32768-61000.

```
echo “32768 61000” > /proc/sys/net/ipv4/ip_local_port_range
```

Add the above commands to */etc/rc.d/rc.local* file and you’ll not have to type it again the next time if you reboot your system.

Installed files

```
/usr/sbin/apxs
/usr/sbin/ab
/usr/sbin/httpd
/usr/sbin/logresolve
/usr/sbin/rotatelogs
/usr/bin/dbmmanage
/usr/bin/htdigest
/usr/bin/htpasswd
/usr/include/apache/
/usr/lib/apache/
/etc/httpd/
/etc/logrotate.d/apache
/etc/rc.d/init.d/httpd (chkconfig --del httpd)
/home/httpd/cgi-bin/
/home/httpd/htdocs/
/usr/man/man1/dbmmanage.1
/usr/man/man1/htdigest.1
/usr/man/man1/htpasswd.1
/usr/man/man8/ab.8
/usr/man/man8/apxs.8
/usr/man/man8/httpd.8
/usr/man/man8/logresolve.8
/usr/man/man8/rotatelogs.8
/var/log/httpd/
/var/cache/
```

Optional component to install with Apache

Devel-Symdump

The perl module Devel::Symdump provides a convenient way to inspect perl's symbol table and the class hierarchie within a running program. From version 2.00, this module needs at least perl5.003. To build and install it, please follow this step.

Packages
Devel-Symdump Homepage: http://www.perl.com/CPAN/modules/by-module/Devel/

```
[root@deep]# cp Devel-Symdump-version.tar.gz /var/tmp/
[root@deep]# tar xzpf Devel-Symdump-version.tar.gz
```

Cd into the new devel-Symdump directory and type the following commands on your terminal:

```
[root@deep]# perl Makefile.PL
[root@deep]# make
[root@deep]# make test
[root@deep]# make install
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf devel-Symdump.version/ Devel-Symdump-version.tar.gz
```

CGI.pm
This is CGI.pm, an easy-to-use Perl5 library for writing World Wide Web CGI scripts. Older version of this software exists by default on your system and are buggy, please update to version 2.51 at least. To install this module, please follow this step.

Packages
CGI.pm Homepage: http://stein.cshl.org/WWW/software/CGI/cgi_docs.html

```
[root@deep]# cp CGI.pm.tar.gz /var/tmp/
[root@deep]# tar xzpf CGI.pm.tar.gz
```

Cd into the new CGI.pm directory and type the following commands on your terminal:

```
[root@deep]# perl Makefile.PL
[root@deep]# make
[root@deep]# make test
[root@deep]# make install
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf CGI.pm/ CGI.pm.tar.gz
```

Formmail
FormMail is a universal WWW form to E-mail gateway. There is only one required form input tag, which must be specified in order for this script to work with your existing forms. To install this script, please follow this step.

Packages
Formmail Homepage: http://www.worldwidemart.com/scripts/formmail.shtml

```
[root@deep]# cp formmail.tar.gz /var/tmp/
[root@deep]# tar xzpf formmail.tar.gz
```

Cd into the new Formmail directory and edit the following file:

```
[root@deep]# vi Formmail.pl
```

@referers = (**'openarch.com','192.168.1.1'**);
This array allows you to define the domains that you will allow forms to reside on and use your FormMail script.

The script, FormMail.pl, needs to be placed in your server's cgi-bin and the www user must have the ability to read/execute the script.

[root@deep]# cp Formmail.pl /home/httpd/cgi-bin/ (or wherever your CGIs live).
Cd to cgi-bin directory
[root@deep]# chmod 750 Formmail.pl
[root@deep]# chown 0.99 Formmail.pl

Cleanup after work
[root@deep]# cd /var/tmp
[root@deep]# rm -rf formmail/ formmail.tar.gz

Webalizer
The Webalizer is a web server log file analysis program, which produces usage statistics in HTML format for viewing with a browser. The results are presented in both columnar and graphical format, which facilitates interpretation.

Packages
Webalizer Homepage: http://www.mrunix.net/webalizer/

[root@deep]# cp webalizer-version-src.tgz /var/tmp/
[root@deep]# tar xzpf webalizer-version-src.tgz

The Webalizer requires the GD graphics library by Tom Boutell. If you don't already have it, install it from the Red Hat Linux 6.1 CD-ROM.

[root@deep]# rpm –Uvh gd-devel-version.i386.rpm

Cd into the new Webalizer directory and type the following commands on your terminal:

```
CC=”egcs” \
CFLAGS=”-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer –fno-exceptions” \
./configure \
--prefix=/usr
[root@deep]# make
[root@deep]# make install
[root@deep]# mkdir /home/httpd/usage
```

Add the following lines in your httpd.conf file:

```
Alias /usage/ "/home/httpd/usage/"
 <Directory "/home/httpd/usage">
        Order deny,allow
        Deny from all
        Allow from 192.168.1.3
</Directory>
```

Cleanup after work
[root@deep]# cd /var/tmp
[root@deep]# rm –rf webalizer-version/ webalizer-version-src.tgz

FAQ-O-Matic

The Faq-O-Matic is a CGI-based system that automates the process of maintaining a FAQ (or Frequently Asked Questions list). It allows visitors to your FAQ to take part in keeping it up-to-date. To install this program, please follow this step.

Packages
FAQ-O-Matic Homepage: http://www.dartmouth.edu/~jonh/ff-serve/cache/1.html

```
[root@deep]# cp FAQ-OMatic-version.tar.gz /var/tmp/
[root@deep]# tar xzpf FAQ-OMatic-version.tar.gz
```

Cd into the new FAQ-O-Matic directory and type the following commands on your terminal:

```
[root@deep]# perl Makefile.PL
[root@deep]# make
[root@deep]# make install
[root@deep]# mv fom /home/httpd/cgi-bin/ (or wherever your CGIs live).
[root@deep]# mkdir -p /home/httpd/cgi-bin/fom-meta
[root@deep]# mkdir -p /home/httpd/faqomatic
[root@deep]# chown 0.99 /home/httpd/cgi-bin/fom
[root@deep]# chown -R 99.99 /home/httpd/cgi-bin/fom-meta/
[root@deep]# chown -R 99.99 /home/httpd/faqomatic/
```

Add the following lines in your httpd.conf file:

```
Alias /faqomatic/ "/home/httpd/faqomatic/"
<Directory "/home/httpd/faqomatic">
        Order allow,deny
        Allow from all
</Directory>

Alias /bags/ "/home/httpd/faqomatic/bags/"
<Directory "/home/httpd/faqomatic/bags">
        Order allow,deny
        Allow from all
</Directory>

Alias /cache/ "/home/httpd/faqomatic/cache/"
<Directory "/home/httpd/faqomatic/cache">
        Order allow,deny
        Allow from all
</Directory>

Alias /item/ "/home/httpd/faqomatic/item/"
<Directory "/home/httpd/faqomatic/item">
        Order allow,deny
        Allow from all
</Directory>
```

Restart you webserver with the following command:

```
[root@deep]# /etc/rc.d/init.d/httpd restart
```

Netscape http://localhost/cgi-bin/fom (or whatever browser you prefer). (Or whatever the URL would be to execute the CGI).
Enter your temporary password
Create the fom-meta directory
Click first on "**Define configuration parameters**" and configure
Configure the following for example under: "**Mandatory**: Server directory configuration"
$serverBase= http://www.openarch.com
$cgiURL= /cgi-bin/fom
$serveDir= /home/httpd/faqomatic/
$serveURL= /faqomatic/

configure the rest as you need

Cleanup after work
[root@deep]# cd /var/tmp
[root@deep]# rm -rf FAQ-OMatic-version/ FAQ-OMatic-version.tar.gz

Webmail IMP
IMP is an IMAP Webmail client.
To install this program, please follow this step.

Packages
Webmail IMP Homepage: http://www.horde.org/imp/

[root@deep]# cp horde-version.tar.gz /home/httpd/
[root@deep]# tar xzpf horde-version.tar.gz
[root@deep]# mv horde-version horde
[root@deep]# cp imp-version.tar.gz /home/httpd/horde/
Change into the “horde” directory (cd /home/httpd/horde/), and untar/gzip imp-version.tar.gz
[root@deep]# tar xzpf imp-version.tar.gz
[root@deep]# rm -f imp-version.tar.gz
[root@deep]# mv imp-version imp
Add the following lines in your httpd.conf file:

```
Alias /horde/ "/home/httpd/horde/"
<Directory "/home/httpd/horde">
        Order allow,deny
        Allow from all
</Directory>

Alias /imp/ "/home/httpd/horde/imp/"
<Directory "/home/httpd/horde/imp">
        Options None
        Order allow,deny
        Allow from all
</Directory>
```

Restart you webserver with the following command:
[root@deep]# **/etc/rc.d/init.d/httpd restart**

New setup engine named “setup.php3” give people the ability to configure IMP via the web. For security reasons, it is disabled by default, but you can enable if by saying:

Cd into horde directory (cd /home/httpd/horde/) and type the following on your terminal
[root@deep]# sh ./install.sh

You should be able to point your browser to http:// <your imp server>/<your horde home>/setup.php3. At this point you can walk through the graphical setup program and configure all aspects of IMP.

When you are done be sure to disable it again!
Cd horde directory (cd /home/httpd/horde/) and type the following on your terminal
[root@deep]# sh ./secure.sh

Cleanup after work
[root@deep]# cd /var/tmp
[root@deep]# rm -rf horde-version.tar.gz

Linux Tripwire

Overview

With the advent of increasingly sophisticated and subtle account break-ins on Unix systems, the need for tools to aid in the detection of unauthorized modification of files becomes clear. Tripwire is a tool that aids system administrators and users in monitoring a designated set of files for any changes. Used with system files on a regular (e.g., daily) basis, Tripwire can notify system administrators of corrupted or tampered files, so damage control measures can be taken in a timely manner. Tripwire is a file and directory integrity checker, a utility that compares a designated set of files and directories against information stored in a previously generated database. Any differences are flagged and logged, including added or deleted entries. When run against system files on a regular basis, any changes in critical system files will be spotted -- and appropriate damage control measures can be taken immediately. With Tripwire, system administrators can conclude with a high degree of certainty that a given set of files remain free of unauthorized modifications if Tripwire reports no changes.

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
Tripwire version number is 1.3.1-1

Packages

Tripwire Homepage: http://www.tripwiresecurity.com/

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install it, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filetrip1.txt**’ before and ‘**find /* > filetrip2.txt**’ after you install the tarball, and use ‘**diff filetrip1.txt filetrip2.txt > difftrip.txt**’ to get a list of what changed.

Compilation Tripwire-1.3.1-1

Decompress the tarball (tar.gz).

```
[root@deep]# cp Tripwire-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf Tripwire-version.tar.gz
```

Compile and Optimize

Cd into the new Tripwire dir and type the following on your terminal:

[root@deep]# vi +462 src/utils.c and change:

```
else if (iscntrl(*pcin)) {
```

To:

else if (!(*pcin & 0x80) && iscntrl(*pcin)) {

[root@deep]# vi +356 src/config.parse.c and change:

```
rewind(fpout);
To:
    else {
        rewind(fpin);
     }
```

[root@deep]# vi +106 include/config.h and change:

```
#define CONFIG_PATH     "/usr/local/bin/tw"
#define DATABASE_PATH   "/var/tripwire"
To:
#define CONFIG_PATH     "/etc"
#define DATABASE_PATH   "/var/spool/tripwire"
```

[root@deep]# vi +165 include/config.h and change:

```
#define TEMPFILE_TEMPLATE "/tmp/twzXXXXXX"
To:
#define TEMPFILE_TEMPLATE "/var/tmp/.twzXXXXXX"
```

[root@deep]# vi +66 src/config.pre.y and change:

```
#ifdef TW_LINUX
To:
#ifdef TW_LINUX_UNDEF
```

[root@deep]# vi +13 Makefile and change:

```
DESTDIR = /usr/local/bin/tw
To:
DESTDIR = /usr/sbin

DATADIR = /var/tripwire
To:
DATADIR = /var/spool/tripwire

LEX     = lex
To:
LEX     = flex

CC=gcc
To:
CC=egcs

CFLAGS = -O
To:
CFLAGS = -O6
```

```
[root@deep]# make
[root@deep]# make install

[root@deep]# chmod 700 /var/spool/tripwire/
[root@deep]# chmod 500 /usr/sbin/tripwire
[root@deep]# chmod 500 /usr/sbin/siggen
[root@deep]# rm -f /usr/sbin/tw.config
```

Cleanup after work

```
[root@deep]# cd /var/tmp
```

```
[root@deep]# rm -rf tw_ASR_version/ Tripwire-version.tar.gz
```

Configurations

Configuration files for different services are very specific depending of your need and your network architecture. Someone can install Samba Server and have just one client connection and other can install it with 1000 connections. People can install DNS Server at home like a caching only DNS and company can install it with primary and secondary DNS servers. For these reasons, we'll focus on optimization and security of these files and let all specific adjustments to your tastes. So you will need to read documentation that comes with these software and understand them. All server software we describe on our documentation "Linuxsos.pdf" will have a specific directory and subdirectory in the tar compressed file "floppy.tgz" containing file configuration for the specific server. It will be to your responsibility to place these files to the appropriated places on your server machine, like show bellow. Server Configuration files archive are located at http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

Copy **tw.config** file to */etc/* directory (tw.config must be 600).
Copy **tripwire.verify** script to */etc/cron.daily/* directory (tripwire.verify must be 700).

You can obtain configuration files listed bellow on our floppy.tgz archive. Copy the following files from the decompressed floppy.tgz archive to their appropriated place.

Configuration of the /etc/tw.config file

The tw.config file define all the directories that contain files that you wan monitories.

Create the **tw.config** file (touch /etc/tw.config) and add:

```
# Gerhard Mourani: gmourani@netscape.net
# last updated: 1999/11/12

#  First, root's "home"
/root                   R
/                       R
!/root/tmp
!/cache

# OS itself
/boot/vmlinuz           R

# critical boot resources
/boot                   R

# Critical directories and files
/chroot                 R
/etc                    R
/etc/inetd.conf         R
/etc/nsswitch.conf      R
/etc/rc.d               R
/etc/mtab               L
/etc/motd               L
/etc/group              R
/etc/passwd             L

# other popular filesystems
/usr                    R
/usr/local              R
/dev                    L-am
/usr/etc                R
```

```
# truncate home
=/home                  R

# var tree
=/var/spool             L
/var/log                L
/var/lib                L
/var/spool/cron         L
!/var/lock

# unusual directories
=/proc                  E
=/tmp
=/mnt/cdrom
=/mnt/floppy
```

Make the mode of this file 600 with the following command:
[root@deep]# **chmod 600 /etc/tw.config**

Configuration of the /etc/tripwire.verify script

The tripwire.verify file is a small script executed by the crond program each days to scan your hard disk for changed files or directories.

Create the **tripwire.verify** script (touch /etc/cron.daily/tripwire.verify) and add:

```
#!/bin/sh
/usr/sbin/tripwire -loosedir -q | (cat <<EOF
This is an automated report of possible file integrity changes, generated by
the Tripwire integrity checker. To tell Tripwire that a file or entire
directory tree is valid, as root run:

/usr/sbin/tripwire -update [pathname|entry]

If you wish to enter an interactive integrity checking and verification
session, as root run:

/usr/sbin/tripwire -interactive

Changed files/directories include:
EOF
cat
) | /bin/mail -s "File integrity report" root
```

Make this script executable and mode 700 with the following command:
[root@deep]# **chmod 700 /etc/cron.daily/tripwire.verify**

Commands

The commands listed bellow are some that we use often in our regular use but much more exist and you must check the man page for more details and information.

Creating file information database, which is used for all subsequent Integrity Checking runs.

[root@deep]# **cd /var/spool/tripwire/**
[root@deep]# **/usr/sbin/tripwire --initialize**

Interactive Update mode. Tripwire first reports all added, deleted, and changed files, then allows the user to update the entry in the database.

```
[root@deep]# cd /var/spool/tripwire/database/
[root@deep]# cp tw.db_myserverhostname /var/spool/tripwire/
[root@deep]# cd ..
[root@deep]# /usr/sbin/tripwire --interactive
```

Installed files

/var/spool/tripwire/
/etc/tw.config
/etc/cron.daily/tripwire.verify
/usr/sbin/tripwire
/usr/sbin/siggen
/usr/man/man5/tw.config.5
/usr/man/man8/tripwire.8
/usr/man/man8/siggen.8

Security Issue: For added security, it is recommended that clear text copies of policy and configuration files not be stored on systems where Tripwire is deployed. You may also wish to back up Tripwire files on secondary media or to store them in a remote location for additional safety.

Linux GnuPG

Overview

GnuPG - The GNU Privacy Guard
GnuPG is GNU's tool for secure communication and data storage. It can be used to encrypt data and to create digital signatures. It includes an advanced key management facility and is compliant with the proposed OpenPGP Internet standard as described in RFC2440. Because GnuPG does not use any patented algorithm it cannot be compatible with PGP2 versions. PGP 2.x uses only IDEA (which is patented worldwide) and RSA (which is patented in the United States until Sep 20, 2000).

These installation instructions assume

Commands are Unix-compatible.
The source path is /var/tmp (other paths are possible).
Installations were tested on RedHat Linux 6.1.
All steps in the installation will happen in superuser account “root”.
GnuPG version number is 1.0.0

Packages

GnuPG Homepage: http://www.gnupg.org/

Tarballs

I would advise using tarballs, it is a good idea to make a list of files on the system before you install it, and one afterwards, and then compare them using ‘**diff**’ to find out what file it placed where. Simply run ‘**find /* > filepg1.txt**’ before and ‘**find /* > filepg2.txt**’ after you install the tarball, and use ‘**diff filepg1.txt filepg2.txt > diffpg.txt**’ to get a list of what changed.

Compilation

Decompress the tarball (tar.gz).

```
[root@deep]# cp gnupg-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf gnupg-version.tar.gz
```

Compile and Optimize

Cd into the new GnuPG dir and type the following on your terminal:

```
CC="egcs" \
CFLAGS="-O6 -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--enable-shared

[root@deep]# make
[root@deep]# make check
[root@deep]# make install
[root@deep]# strip /usr/bin/gpg
```

Cleanup after work

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf gnupg-version/ gnupg-version.tar.gz
```

Commands

The commands listed bellow are some that we use often in our regular use but much more exist and you must check the man page for more details and information.

1) To encrypt and sign data, use the following command:

```
[root@deep]# gpg -e --sign --armor <mymessage>
```

For example:

```
[root@deep]# gpg -e --sign --armor message-to-paul.txt
You need a passphrase to unlock the secret key for
user: "Gerhard Mourani (Open Network Architecture) <gmourani@videotron.ca>"
1024-bit DSA key, ID BBB4BA9B, created 1999-10-26
Enter passphrase:
```

-Which mean “-e” is for encrypting, “--sign” for signing, “--armor” for create ASCII armored output and mymessage is the message you wan to encrypt.

2) To decrypt data, use the following command:

```
[root@deep]# gpg -d <mymessage.asc>
```

For example:

```
[root@deep]# gpg -d message-to-admin.asc
You need a passphrase to unlock the secret key for
user: "Gerhard Mourani (Open Network Architecture) <gmourani@videotron.ca>"
2048-bit ELG-E key, ID 71D4CC44, created 1999-10-26 (main key ID BBB4BA9B)
Enter passphrase:
```

-Which mean “-d” is for decrypting and <mymessage.asc> is the message you wan to decrypt.

3) To extract your public key in ASCII armored output, use the following command:

[root@deep]# gpg --export --armor > Public-key.asc

-Which mean "—export" is for extracting your Public-key from your pubring encrypted file, "--armor" for create ASCII armored output that you can mail or publish and "> Public-key.asc" to put the result in a file that you're named Public.key.asc.

4) To sign a message or a binary file, use the following command:

[root@deep]# gpg --detach-sign --armor <Data>

-You can write the signature in a separate file. It is highly recommended to use this option especially when signing binary files (like archives for instance). Also the --armor option can be extremely useful here. <data> can be a file or a binary file like tar archive.

5) To check the signature of encrypted data, use the following command:

When encrypted data has been signed as well, the signature is checked when the data is decrypted. You can check the signature of signed data by using the command:

[root@deep]# gpg --verify <Data>

This will only work (of course) when you own the public key of the sender. <data> is the same above.

Installed files

/usr/bin/gpg
/usr/lib/gnupg
/usr/lib/gnupg/rndunix
/usr/lib/gnupg/rndegd
/usr/lib/gnupg/tiger
/usr/man/man1/gpg.1
/usr/share/locale/de/LC_MESSAGES/gnupg.mo
/usr/share/locale/es_ES/LC_MESSAGES/gnupg.mo
/usr/share/locale/fr/LC_MESSAGES/gnupg.mo
/usr/share/locale/pl/LC_MESSAGES/gnupg.mo
/usr/share/locale/pt_BR/LC_MESSAGES/gnupg.mo
/usr/share/locale/it/LC_MESSAGES/gnupg.mo
/usr/share/locale/ru/LC_MESSAGES/gnupg.mo
/usr/share/gnupg
/usr/share/gnupg/options.skel

Linux IPX Netware ™

Overview

Internet Packet exchange is a protocol used by the Novell corporation to provide internetworking support for their NetWare™ product.

Ncpfs is a filesystem, which understands the Novell NetWare(TM) NCP protocol. Functionally, NCP is used for NetWare the way NFS is used in the TCP/IP world. For a Linux system to mount a NetWare filesystem, it needs a special mount program. The ncpfs package contains such a mount program plus other tools for configuring and using the ncpfs filesystem. You must verify that ncpfs is installed on your system. Use the command **rpm –q ncpfs**.

The **ipxutils** package includes utilities (ipx_configure, ipx_internal_net, ipx_interface, and ipx_route) necessary for configuring and debugging IPX interfaces and networks under Linux. IPX is the low-level protocol used by Novell's NetWare file server system to transfer data. You must verify that ipxutils is installed on your system. Use the command **rpm –q ipxutils**.

These installation instructions assume

Commands are Unix-compatible.
Installations were tested on RedHat Linux 6.1 Server.
All steps in the installation will happen in superuser account “root”.
ncpfs version number is 2.2.0.12
ipxutils version number is 2.2.0.12

Build a kernel with IPX support and NCP protocol

The first thing you need to do is ensure that your kernel has been built with IPX support enabled and NCP protocol. In the 2.2.13 kernel version you need ensure that you have answered Y to the following question:

The IPX protocol (CONFIG_IPX) [N] Y
NCP filesystem support (CONFIG_NCP_PS) [N] Y
Packet signatures (CONFIG_NCPFS_PACKET_SIGNING) [N] Y
Clear remove/delete inhibit when needed (CONFIG_NCPFS_STRONG) [N] Y
Use NFS namespace if available (CONFIG_NCPFS_NFS_NS) [N] Y
Use LONG (OS/2) namespace if available (CONFIG_NCPFS_OS2_NS) [No] Y
Allow mounting of volume subdirectories (CONFIG_NCPFS_MOUNT_SUBDIR) [N] Y
Enable symbolic links and execute flags (CONFIG_NCPFS_EXTRAS) [N] Y

Trying to set up an IPX only network interface with no TCP/IP

You can have the interface active without any protocols bound to it. Instead of using ifconfig to place IP numbers, etc. Simply say ifconfig ethN up

Assuming you want to setup eth1 as an IPX only network interface without TCP/IP, you must check that the following file exist (ifcfg-eth1) and be sure the lines IPADDR, NETMASK, NETWORK and BROADCAST contain no values:

[root@deep]# vi /etc/sysconfig/network-scripts/ifcfg-eth1

DEVICE=eth1
IPADDR=
NETMASK=
BROADCAST=
ONBOOT=no
BOOTPROTO=none
USERCTL=no

After, all you have to do is to restart the network daemon with the command: **/etc/rc.d/init.d/network restart** and make up the Ethernet card eth1 with the command**: ifconfig eth1 up**.

Since Linux by default configure the IPX protocol on both interfaces (eth0 and eth1) and make the first card it found the primary IPX interface (eth0) you must to deactivate the IPX protocol on this card (eth0) because we want to use our second card (eth1) to transfer IPX data's.

To make this, use the command: **ipx_interface del eth0 802.3**. Now if you make an ifconfig, you will see that our network interfaces are configured in this way: eth0 TCP/IP protocol only and eth1 IPX protocol only.

Ncpfs User Commands

For automatic setting of the interface configuration and primary interface, use the command
[root@deep]# **ipx_configure –auto_interface=on –auto_primary=on**

To mount a Novell™ server or volume, use the command
[root@deep]# **ncpmount –S DMS01 /mnt/netware –U username –P passwd**

This command will mount fileserver DMS01, with a login id of username with password passwd, under the /mnt/netware directory. If you don't specify the –P option you will be prompted for a password.

To umount the /mnt/netware directory, use the command
[root@deep]# **ncpumount /mnt/netware**

To see a list of all of the Novell fileserver on your network, use the command
[root@deep]# **slist**

To copy files, use the command
[root@deep]# **ncopy file1 [file2…] directory**

There are a number of files related to the Linux IPX support that are located within the /proc filesystem. They are:

/proc/net/ipx_interface
This file contains information about the IPX interfaces configured on your machine. These may have been configured manually by command or automatically detected and configured.

/proc/net/ipx_route
This file contains a list of the routes that exist in the IPX routing table. These routes may have been added manually by command or automatically by an IPX routing daemon.

/proc/net/ipx
This file is a list of the IPX sockets that are currently open for use on the machine.

Linux FTP Server

Overview

Using the File Transfer Protocol (FTP) is a popular way to transfer files from machine to machine across a network. Clients and servers have been written for all the popular platforms, thereby often making FTP the most convenient way of performing file transfers.

You can configure FTP servers in one of two ways. The first is as a private user-only site, which is the default configuration for the FTP server; I will cover this configuration here. A private FTP server allows users on the system only to be able to connect via FTP and access their files. You can place access controls on these users so that certain users can be explicitly denied or granted access.

The other kind of FTP server is anonymous. An anonymous FTP server allows anyone on the network to connect to it and transfer files without having an account. Due to the potential security risk involved with this setup, you should take precautions to allow access only to certain directories on the system.

The **wu-ftpd** package contains the wu-ftpd FTP (File Transfer Protocol) server daemon. The FTP protocol is a method of transferring files between machines on a network and/or over the Internet. Wu-ftpd's features include logging of transfers, logging of commands, on the fly compression and archiving, classification of users' type and location, per class limits, per directory upload permissions, restricted guest accounts, system wide and per directory messages, directory alias, cdpath, filename filter and virtual host support. You must verify that wu-ftpd is installed on your system. Use the command **rpm –q wu-ftpd**.

These installation instructions assume

Commands are Unix-compatible.
Installations were tested on RedHat Linux 6.1 Server.
All steps in the installation will happen in superuser account “root”.
Wu-ftpd version number is 2.6.0

Packages:

Wu-ftpd Homepage: http://www.wu-ftpd.org/

How the FTP Server Works

FTP service is controlled from the ***/etc/inetd.conf*** file and is automatically invoked whenever someone connects to the FTP port. When a connection is detected, the FTP daemon (/usr/sbin/in.ftpd) is invoked and the session begins. In the /etc/inetd.conf file, the default Red Hat distribution contains the necessary line for this step to occur.

Users accessing the FTP server are placed in their home directories when they first log in. At that point, they can change into any directories on the system to which they have permission. Anonymous users, on the other hand, have several restrictions placed on them.

Configuring the FTP Server

Although the default configuration of the FTP server is reasonably secure, you can fine-tune access rights by editing the following files:

- /etc/ftpaccess
- /etc/ftpconversions
- /etc/ftphosts
- /var/log/xferlog (log file for the FTP server)

With all these files, you can have very fine control of who, when, and from where people can connect to your server as well as an audit trail of what they did after they did connect. The ***/etc/ftpaccess*** file is the most significant of these because it contains the most configuration options; however, misconfiguring any of the others can lead to denied service.

The /etc/ftpaccess file

The /etc/ftpaccess file is the primary means of controlling who and how many users access your server. Each line in the file controls either defines an attribute or sets its value. The following commands define the criteria used to determine in which group each user should be placed.

class

The class command defines a class of users who can access your FTP server. You can define as many classes as you want. Each class line comes in the form

class <*classname*> <*typelist*> <*addrglob*>

-where <*classname*> is the name of the class you are defining, <*typelist*> is the type of user you are allowing into the class, and <*addrglob*> is the range of IP addresses allowed access to that class.

The <*typelist*> is a comma-delimited list in which each entry has one of three values: **anonymous, guest, or real**. Anonymous users are, of course, any users who connect to the server as user anonymous or ftp and want to access only publicly available files. Guest users are special because they do not have accounts on the system per se, but they do have special access to key parts of the guest group. Real users must have accounts on the FTP server and are authenticated accordingly.

<*addrglob*> takes the form of a regular expression where * implies all sites.

The line **class all real 207.164.186.***

allows only real users with accounts on the FTP server access to their accounts via FTP if they are coming from the address 207.164.186.*

deny

The deny command allows you to explicitly deny service to certain host based on either their names, IP addresses, or whether their host's names can be reverse-resolved via DNS. The format of the deny command is

deny <*addrglob*> <*message_file*>

-where <*addrglob*> is a regular expression containing the addresses that are to be denied and <*message_file*> is the filename containing a message that should be displayed to the hosts when they connect.

The following is a sample deny line:

deny evilhacker.domain.com /home/ftp/.deny.msg

this line display the contents of the file /home/ftp/message.no.evil.hackers to anyone trying to connect via FTP from evilhacker.domain.com.

limit
The limit command allows you to control the number of users who log in to the system via FTP by class and time of day. This capability is especially useful if you have a popular archive but the system needs to be available to your users during business hours. The format of the limit command is

limit <*class*> <*n*> <*times*> <*message_file*>

-where <*class*> is the class to limit, <*n*> is the maximum number of people allowed in that class, <*times*> is the time during which the limit is in effect, and <*message_file*> is the file that should be displayed to the client when the maximum limit is reached.

The format of the <*times*> parameter is somewhat complex. The parameter is in the form of a comma-delimited string, where each option is for a separate day. Sunday through Saturday take the form Su, Mo, Tu, We, Th, Fr, and Sa, respectively, and all the weekdays can be referenced as Wk. Time should be kept in military format without a colon separating the hours and minutes. The dash character specifies a range.

For example, to limit the class **all** to 10 users from Monday through Thursday, all day, and Friday from midnight to 5:00 p.m., you would use the following limit line:

limit all 10 MoTuWeTh,Fr0000-1700 /home/ftp/.too_many.msg

in this case, if the limit is hit, the contents of the file /home/ftp/.message.too.many are displayed to the connecting user.

loginfails
The loginfails command allows you to set the number of failed login attempts clients can make before disconnecting them. By default, this number is five; however, you can set it by using the command

loginfails <*n*>

-where <*n*> is the number of attempts. For example, the following line disconnects a user from the FTP server after three failed attempts:
loginfails 3

message
the message command allows you to set up special messages to be sent to the clients when they either log in or change into a certain directory. You can specify multiple messages. The format of this command is

message <*path*> <*when*>

-where <*path*> is the full pathname to the file to be displayed, <*when*> is the condition under which to display the message.

The <*when*> parameter should take one of two forms: either LOGIN or CWD=<dir>. If it is LOGIN, the message is displayed upon a successful login. If the parameter is set to CWD=<dir>, then the message id displayed when client enter the <dir> directory.

Remember that when messages are triggered by an anonymous user, the message path needs to be relative to the anonymous FTP directory.

An example message command is

message /welcome.msg LOGIN

readme
The readme command allows you to specify the conditions under which clients are notified that a certain file in their current directory was last modified. This command can take the form

readme <*path*> <*when*>

-where <*path*> is the name of the file to alert the clients about (for example README), <*when*> is similar to the <*when*> in the message command.

Remember that when you're specifying a path for anonymous users, the file must be relative to the anonymous FTP directory.

log transfers
If you want to log only file transfers made by clients instead of their entire sessions with the log commands statement, you should use log transfers instead. The format of this command is

log transfers <*typelist*> <*directions*>

-where <*typelist*> is a comma-separated list specifying which kinds of users should be logged (**anonymous, guest, real**), and <*directions*> is a comma-separated list specifying which direction the transfers must take in order to be logged. The two directions you can choose to log are inbound and outbound.

For example, to log all **real** transfers that are both inbound and outbound, you would use the following:

log transfers real inbound,outbound

The resulting logs are stored in /var/log/xferlog.

upload
You can use the upload command, to control files placed onto your server. The upload command specifies what permissions the client has to place files in certain directories as well as what permissions the files will take on after they are placed there. The format of this command is

upload <*directory*> <*dirglob*> <*switch*> <*owner*> <*group*> <*mode*> <*mkdir*>

-where <*directory*> is the directory that is affected by this command, <*dirglob*> is the regular expression used to determine whether a subdirectory under <*directory*> is a valid place to make an upload, and <*switch*> is either YES or NO, thereby establishing either an upload can or cannot occur there. The <*owner*>, <*group*>, and <*mode*> parameters establish the file's owner, group, and permissions after the file is placed on the server. Finally, you can specify the <*mkdir*> option as either dirs or nodirs, which allows the client to able to create or not create subdirectories under the specified directory.

Here is a sample entry:

upload /home/ftp /incoming yes ftp ftp 0400 nodirs

this example specifies that the only location a file can be placed is in the /home/ftp/incoming directory (/incoming to the anonymous client). After the file is placed in this directory, its owner becomes ftp, group becomes ftp, and the permission is 0400. The nodirs option at the end of the second line doesn't allow the anonymous client to create subdirectories under /incoming.

The /etc/ftphosts file

The /etc/ftphosts file establishes rule on a per-user basis defining whether users are allowed to log in from certain hosts or whether users are denied access when they try to log in from other hosts.

Each line in the file can be one of two commands:

allow <*username*> <*addrglob*>
or
deny <*username*> <*addrglob*>

-where the allow command allows the user specified in <*username*> to connect via FTP from the explicitly listed addresses in <*addrglob*>. You can list multiple addresses.

The deny command explicitly denies the specified user <*username*> from the sites listed in <*addrglob*>. You can list multiple sites.

FTP Administrative Tools

ftpwho

ftpwho displays all active users on the system connected through FTP. The output of the command is in the format of the /bin/ps command. The format of this command is

<*pid*> <*time*> <*tty*> <*connection details*>

-where <*pid*> is the process ID of the FTP daemon handling the transfer; <*time*> indicates when the user have been connected to the FTP server; <*tty*> is always a question mark (?) because the connection is coming from FTP, not Telnet; and finally, <*connection details*> tells where the connection is coming from, who is the user, and what that user,s current function is.

The following is an example of output from ftpwho:

```
Service class all:
webmaste  2773   337  0 09:49 ?        00:00:00 ftpd: gate.openarch.com: webmast
  -  1 users ( 10 maximum)
```

Here, you can see that one user is logged in (10 users are allowed to connect) and this user has the username webmaste who claims to be gate.openarch.com.

ftpcount

ftpcount, which is a simplified version of ftpwho, shows only the total count of users logged in to the system and the maximum number of users allowed. A sample output from ftpcount shows the following:

```
Service class all              -  1 users ( 10 maximum)
```

Securing FTP

ENSURE that you have set up a file ***/etc/ftpusers*** which specifies those users that are NOT allowed to connect to your ftpd. This should include, as a MINIMUM, the entries: root, bin, uucp, daemon, adm, lp, sync, shutdown, halt, mail, news, operator, games, nobody, xfs and ALL vendor supplied accounts.

To disable anonymous ftp, move or delete all files in ***/home/ftp/*** and then remove the user **ftp** from your password file. Verify that **anonftp-version.i386.rpm** package is not installed on your system whit the command [root@deep]# **rpm –q anonftp**.

noretrieve
Access capability "**noretrieve**" is a parameter that you configure in your */etc/ftpaccess* file. This command denies retrieve-ability of specific files or directories. If the files are an absolute path specification (i.e. begins with '/' character) then only those files are marked un-gettable, otherwise all files with matching the filename are refused transfer. Example:

noretrieve /etc/passwd core

-specifies no one will be able to get the file */etc/passwd* whereas they will be allowed to transfer a file 'passwd' if it is not in /etc. On the other hand no one will be able to get files named 'core' wherever it is.

Absolute path specifications ending with a slash ('/') are taken to mean all files in the named directory are un-gettable. Example:

noretrieve /etc/
noretrieve /usr/
noretrieve /var/

-specifies no one will be able to get any files on */etc/* or */usr/* or */var/* directories.

log security
Enables logging of violations of security rules (noretrieve, .notar, ...) for real, guest and/or anonymous users. This is a parameter that you configure in your */etc/ftpaccess* file.

log security <*typelist*>

-where <*typelist*> is a comma-separated list of any of the keywords "**anonymous**", "**guest**" and "**real**". If the "real" keyword is included, logging will be done for users using FTP to access real accounts, and if the "anonymous" keyword is included logging will done for users using anonymous FTP. The "guest" keyword matches guest access accounts (see "guestgroup" for more information).

restricted-uid, restricted-gid and guest-root
These clauses control whether or not **real** or **guest** users will be allowed access to areas on the FTP site outside their home directories. This is a parameter that you configure in your */etc/ftpaccess* file. An example of the use of these clauses shows their intended use.

guest-root <*root-dir*>
restricted-uid <*uid-range*>
restricted-gid <*gid-range*>

guest-root /home tarzan jane
restricted-uid tarzan jane
restricted-gid tarzan jane

<*root-dir*> specified the chroot() path for users. Multiple uid ranges may be given on the line. If a guest-root is chosen for the user, the user's home directory in the <*root-dir*>*/etc/passwd* file is used to determine the initial directory and their home directory in the system-wide */etc/passwd* is not used. While both tarzan and jane are chroot'd to */home*, they cannot access each other's files because they are restricted to their home directories.

As with all other ftp access rules, try to use directory and file permissions to backstop the operation of the ftpaccess configuration.

greeting full|brief|terse and greeting text
Allows you to control how much information is given out before the remote user logs in. This is a parameter that you configure in your */etc/ftpaccess* file.

greeting *full*|*brief*|*terse*
greeting text <*message*>

-*'greeting full'* is the default and shows the hostname and daemon version. *'greeting brief'* whose shows the hostname. *'greeting terse'* simply says "FTP server ready." The *'text'* form allows you to specify any greeting message you desire. <*message*> can be any string; whitespace (spaces and tabs) is converted to a single space.