Global Information Assurance Certification Paper

Securing a Red Hat Linux 7.2 Anonymous FTP Server with Security Support syslog Server

Brian Melcher

April 2002

for GIAC GCUX Practical Assignment, v1.8, Option 1
GIAC Certified UNIX Security Administrator

The question isn't 'Are you paranoid?'
…It's 'Are you paranoid enough?'

для *Леночкой*,
моя племянница

for *Lenochka*,
my niece

Table of Contents

1 Overview

Security of a computer system is not limited to the boundaries of the system itself. The overall security solution requires security on the part of the system as well as the environment surrounding it. These portions – the individual system, the network topology and infrastructure systems, and its physical location – all have to work in concert with each other. The systems cannot be considered secure without the infrastructure to support it, nor the infrastructure secure without the systems to support it. Security is also not just a matter of configuring a system and network and forgetting about it; it is also a matter of procedures, policies and behaviors to monitor and sustain the system security.

1.1 Procedure Goals

The purpose of this document is to a) present step-by-step procedures to secure an individual system running Red Hat Linux 7.2 for general-purpose use, then b) present step-by-step procedures (additive to the general-purpose server) to configure for an anonymous FTP server, and to c) provide step-by-step procedures (also additive to the general-purpose server procedures) for a security-supportive system – a syslog server.

The general-purpose step-by-step procedures implement security without any specific server purpose in mind, but serves as a foundation for a specific-purpose server. Subsequently, the procedures to build the anonymous FTP server and syslog server are in addition to the steps for building the general-purpose server. The anonymous FTP server is the fundamental goal of this document; the central syslog server is a security-supportive system purposely placed to assist with the security of the FTP server. This security-supportive infrastructure can be utilized for a larger site of systems, not limited to just the anonymous FTP server detailed here.

1.2 Risk Analysis

An anonymous FTP server will be connected to the Internet to provide a wide and varied user community with file upload and download capabilities.

There are several degrees of risk inherent with an anonymous FTP server. The first concern is that it is connected directly to the Internet. Such a system must be protected from unauthorized users and remote probes and attacks, and monitored for probes, attacks and compromises. If an anonymous FTP server allows file uploads, there is the concern that it can be used as a way station for inappropriate or illegal material. To mitigate this, the incoming files will be written to a directory in which immediate file retrieval will not be allowed: all incoming files will be manually relocated by an FTP administrator into a separate FTP-readable directory. Prior to such relocation, the file content should be inspected and verified that it is what it advertises to be. There is also the concern that a normal user with legitimate access to the system could do something while logged into the system that would make the system more vulnerability. To mitigate this concern, anonymous FTP services should be on a system dedicated strictly to providing that service: the normal user community should not have access, even should not have accounts, on the anonymous FTP server.

The security support environment, the syslog server, contributes to the FTP server security in that it records events from the FTP server in a central location for log file perusal in a common location (very helpful if there is more than one FTP servers), and in providing a second copy of the logs for comparison purposes in the event of an incident on the FTP server itself. The syslog server in and of itself has some security risks. The log files located on it are considered as the de-facto, bona fide copy of the log files, and as such, in order to maintain the integrity of the logs, the syslog server must be highly protected and monitored. As with the FTP server, this should not be a user system. As the logs are in a central location, if compromised, the central logs can be altered to conceal the compromise not only for the syslog server, but for all hosts reporting to it. Log files can also consume a great deal of

space, but maintaining a deep log history is also important to recover the sequence of events from an incident long ago. The syslog server must have sufficient disk capacity to maintain the log history, and have an archival process for those log files (such as to tape or write-once media).

Network topology between the anonymous FTP server(s) and the syslog servers is also a point of security consideration. Denial of service attacks can also flood the syslog server with bogus messages, making it less likely to visually see the reported incident embedded in the logs, or drowning out the message altogether such that the syslog server never captures the log. If the syslog server becomes unavailable, all central logging is down. If the network layout is flat – that is, all systems facing the public Internet – both the FTP and syslog servers equally open to probes, scans and attacks. However, if the systems which do not require public Internet access, like the syslog server, could be set aside on a private network not routable through to the Internet, this configuration will isolate these systems from this large external risk. Such a configuration would limit denial of service attacks to the syslog server to just internal threats. However, while this limits problems to just internal threats is a significant security step, it is not a cure-all either; very real threats can originate internally as well.

Additional risk analysis of individual configuration points and choices are part of the discussion of that configuration.

1.3 System Lifetime

Security concerns are not just limited to when the system is initially prepared, but also during the entire system lifetime.

For the purposes of discussion here, system lifetime is categorized into the following phases:

Build: system OS installation, initial configuration (Sections 2 and 3).
Prepare the system with known good media, while isolated from the network.
Verification: testing prior to the deployment (Section 5).
Test to verify the system runs only the configured services, and are configured as as intended.
Deploy: system service installation (Section 6.1).
Monitor: frequent system audits on live system (Section 6.3).
Verify continued correct configuration and operation.
Sustain: system upgrade and maintenance (Sections 6.4 and 6.5).
Keep up to date with the latest security fixes, but test the updates in a test environment first.

1.4 Procedure Philosophies

Due to the generalized nature of the initial step-by-step detailed in Section 2, they can be applied to servers with a variety of end users. Such generality leads itself towards application in a large-scale deployment – many systems with in a site. With such a large quantity of systems, an economy of scale needs to be achieved between system administration and the number of systems being administrated. This economy of scale is in the servers hardware make and model as well as in the operating system, and in the operating systems disk slicing, package selection, package updates and configuration. To assist in achieving this goal, there should be a general sameness about the systems, even if these systems ultimately have different purposes. If the installation and configuration are developed on a layered approach – the general layer initially, then server-specific configurations as additives and/or subtractives to the initial universal layer. With a large degree of sameness, or even complete identicalness, what is implemented and functions on one system can be expected to behave the same on others just like it, achieving an economy of scale.

To put it another way: once the standard has been defined and fully implemented, that solution can be cloned. Make an out right, bit-by-bit copy of that system. An installation done by hand, even by the most conscientious of individuals, is prone to human error. Cloning can be accomplished by a variety

of methods: copying the disk image of an installed system (perhaps using a utility like Ghost, or, if the hardware model has it, hardware mirroring and removable hard drives); making an auto-install CD which installs this specific installation and configuration, or developing scripts for automatic execution following an OS install by KickStart. (KickStart is a utility developed by Red Hat for conducting unattended OS installs; the install options are detailed in a configuration file.) If an implementation functions correctly, cloning that functioning system guarantees each deployed system adheres to exactly the same implementation standard. Cloning turns a manual ad-hoc process into a scalable, repeatable one with repeatable results. It would be unfortunate if a standard were defined that protected against a certain situation, only to find out that it wasn't correctly implemented throughout the site and the systems are vulnerable to it in the end. It is important to consider the 'what' as well as the 'how' of the installation – the configuration being the 'what' and the cloning method the 'how'.

The following items list the philosophies that drive the installation and configuration of the servers:

Host Elements, securing the system—
system minimalization, underlying sameness— disable all services, enable only what is needed.
system integrity checking— ability of the system to compare itself against a standard.
physical system security— physical access control, power & network access, boot media access.

Site Elements—
site log server— system at each site that collects and archives event messages.
site NTP services— provide time synchronization to systems throughout the site.

Out-of-Scope

For this document, the individual computing system described will be a server – it is not a workstation where someone will sit down in front of it and normally log in and conduct their computing activities. Not discussed and expressly out-of-scope for this document are:

Workstation configuration—
console GUI and applications

Server Services—
DNS/DHCP
NIS/NIS+
NNTP
mail server
file server (NFS, AFS, SMB)
Web server

Backups—
Backup and Restore servers
Backup and retention policies

Tuning and optimization—
Settings outside of security

System physical security—

Perimeter defenses—
firewalls
proxy servers

Internal defenses—
inter-site firewalls (between sites, perhaps controlled country access restrictions)
intra-site firewalls (compartmentalization within the site)
proxy servers

The topics and configuration procedures discussed in this document are fundamental to the system and need to be included universally, no matter the purpose of the server. The items not discussed here are services that are not required for basic core system operation and security. These services should not be included universally, but only if the server is to provide said service (e.g. web server, NFS server). These other services, however, do have security concerns of their own.

1.5 Equipment Used

Naming of particular products or brands should not be interpreted as endorsements; they are simply listed to record the products used by the author for development and testing of the procedures documented here.

The procedures detailed within this document were developed and exercised on a variety of systems, all with Red Hat Linux 7.2 running on Intel-based systems. All systems were configured as server systems, regardless of the original intent of the hardware.

IBM 300PL desktops	Pentium, Pentium II and Pentium III models, from 200 to 550MHz with 64, 96 or 128 MB RAM with 4 GB or larger EIDE hard disk drives internal EIDE CD-ROM 10/100 Mbps FastEthernet adapter
IBM 600/600E laptops	Pentium II models, from 300 to 400 MHz with 64 or 128 MB RAM with 4 GB hard disk drives internal EIDE CD-ROM 10/100 Mbps FastEthernet adapter

1.6 Disclaimer

This guide is written in the hopes that it will be useful, practical and educational, but is made without warranty, expressed or implied. Every effort has been made to make this guide as complete and accurate as possible, but no warranty or fitness is implied. This document is by no means a complete description of all that could or should be done to secure an individual system or entire environment. The information herein is provide on an "as is" basis. Use of the content, concepts, and/or examples contained within this document is entirely at your own discretion and should not be substituted for your own awareness and understanding, caution and vigilance.

---

2 System Installation and Configuration

2.1 Hardware Configuration

Before applying the Operating System to the system, there are some Hardware Configuration options to consider to assist with the security and reliability of the server.

2.1.1 BIOS Settings

There are many BIOS features, such as Plug-n-Play and Wake-on-LAN, which are great features when used properly and when the Operating System supports it, but may interfere with the security and/or reliability of the system.

2.1.1.1 Disable Unneeded BIOS Capabilities

The overall principle for hardware settings should be: only enable those features which are known to be needed. In a large-scale deployment of systems, such as the environment used throughout this document, one should not necessarily need such features as PCI hot-swap, because the system components are defined and set during the engineering phase.

2.1.1.2 Set Boot Devices

In the production environment, only allow booting from the local hard drive: do not allow booting from removable media (floppy disks or CD-ROMs) or from the network. For a production system, removing the ability to boot from removable media will harden the system against console-access breaches: in order to boot from the installation media, an attacked would have to have physical console access as well as knowledge on how to re-activate boot ability from removable media.

For development systems, this removable media restriction will prevent the ability from conducting new installations.

2.1.1.3 IBM 300PL BIOS Considerations

Listed below is the IBM 300PL BIOS menu tree to set the BIOS according to the goals stated above. Power-up the system and press <F1> to go into the BIOS, and select the menu options.

> Devices and I/O Ports
>> > USB Setup
>>> > USB Keyboard/Mouse Support [change to DISABLED]
>>> > USB Support [change to DISABLED after keyboard/mouse]

With some Operating Systems with no or incomplete USB support, if using without a keyboard attached or a keyboard switchbox, this system has been observed to hang within 18 to 24 hours of operation while handling the "USB Keyboard check" interrupt handler.

>> > Ethernet Setup
>>> > Ethernet Support [Enabled]
>>> > Network Boot [change to DISABLED]

In a production environment, only allow boots from the local drive.

> Startup Sequence
>> > Primary Startup Sequence [for engineering development]
>>> > First Device: CD-ROM
>>> > Second Device: Diskette Drive
>>> > Third Device: Hard Disk 0
>>> > Fourth Device: Disabled
>> > Primary Startup Sequence [for production deployment]
>>> > First Device: Hard Disk 0
>>> > Second Device: Disabled

> Third Device: Disabled
> Fourth Device: Disabled

> Advanced Setup
> Plug and Play Control
Plug and Play Operating System [NO]

> Power Management
> ACPI BIOS Mode [DISABLED]
APM features should be turned off in the following manner: enable the parent feature, disable the sub-features, then disable the parent feature. Due to known bugs in some of the earlier BIOS releases, even if the parent feature was disabled, the sub-feature enabled setting took precedence.
> APM
> IDE Drives [DISABLED]
> Display [DISABLED]
> Automatic Hardware Power Management [DISABLED]
> APM BIOS Mode [DISABLED]
> Automatic Power On
> Wake on LAN
> Wake on LAN [DISABLED]
In a production environment, if a system is turned off, it should stay off. A rogue "Wake on LAN" signal should not power-on currently off systems.

> Save Settings
> Exit

2.2 Base Operating System Installation

During the Operating System installation, there are several security-related items to consider: package selection, disk partitioning, password encryption method, and initial root password.

2.2.1 Package Selection

The Red Hat installation program allows the installer to select some canned package configurations (workstation, server, laptop) as well as custom option. For maximum control of installation selections, choose the custom install.

a) _____ select custom installation

The Red Hat Linux Operating System is installed by reading and adding Red Hat "RPM" (Red Hat Package Manager) packages. Which packages are chosen depends upon the intended use of the system. If one is not certain that a package should be included, leave that package out and see if the system works – it can always be added in later if it indeed is needed. The packaging system is also good at keeping track of dependencies; if a selected package requires another, it will be added automatically.

Even with a custom install, unless packages are selected individually or KickStart is utilized, the server will likely have installed packages that are unneeded. The entire list of installed packages should be examined; those that are not needed should be removed. The packaging system also checks dependencies with package removal; if others depend upon a package being removed, a message indicating such will be reported and package removal will terminate. There also is the option to force a package removal, regardless of the dependency.

2.2.2 Disk Partitioning

b) _____ partition disk

The Linux system can be partitioned in numerous manners. There is no single "correct" partition scheme, and coming up with a good scheme could be considered more an art than science. What constitutes a good partition scheme will depend upon the purpose of the server, and what is a good scheme for one purpose may ill-fit another. There are some security implications for disk partition that should be taken into account.

2.2.2.1 Initial Partitions

An initial partition scheme should start with the following individual partitions:

```
/
/boot
swap
/home
```

with consideration for separate partitions for:

```
/var
/usr
/usr/local
/tmp
```

/

The system requires a root file system.

/swap

A swap space is necessary for virtual memory. Old UNIX-hands will say swap space should be twice the size of memory, but with modern day hardware with lots of RAM and operating systems with better memory algorithms, this 2x rule is no longer the universal. It is still a good idea to have swap the same size of or twice that of memory, up until about 2GB of swap space. At about this 2GB level, one should consider increasing the amount of physical system memory if that much virtual memory is really required.

/boot

A separate `/boot` partition may also be necessary with dual-boot systems, in order to ensure the boot executables are located within the first 1024 cylinders of the disk.

/home

With this configuration, user home directories are also separated from the core operating system and its configuration and log files. If `/home` were not its own partition and collapsed into the root filesystem, the system would be susceptible to a denial of service vulnerability where a user consumes all the space on the filesystem.

/var

For a server system, which is likely have a good deal of log files, breaking out `/var` into its own file system will similarly prevent log files from filling up the root filesystem.

/usr, /usr/local

Another directory tree to break out would be `/usr`, and possibly also `/usr/local`. When these directories are their own filesystems, the space requirement for root becomes much smaller. Consider mounting `/usr` (and `/usr/local`) in read-only mode to lock-down those file systems. Neither the root nor `/var` filesystems can be mounted read-only.

/tmp

Another directory to consider for it's own partition is `/tmp`. Without `/tmp` defined as its own partition, the root file system can be filled up.

2.2.2.2 Additional Partitions

When considering creating additional partitions, some of the factors to apply are:

- frequency of change of data, frequency of backup need.
  e.g. `/usr` should not need as frequent backups as `/home`.
- likelihood of partition filling up, and would it filling up disrupt operations of the entire machine, or just the specific service?
  e.g. if `/tmp` and `/var` are part of the root partition, `/tmp` could fill up (unintentionally or maliciously) and prevent further logging to `/var`.
- separation of OS and data partitions between physical different drives.
  e.g. user home directories, application data, web pages, database files should be stored on separate drives. With this segregation, the Operating System drive(s) can be treated more as 'fuses' in that they can be replaced with a known good copy in the event of a problem. [This is discussed in more detail later in the 'Backups' Section.]

Additional partitions should be considered depending upon the purpose of the machine:

purpose	critical directory/s	
FTP server	`~ftp, /var/log`	An FTP server may require very little space or lots of it, depending if it's a download-only server or allows for uploads. Red Hat Linux 7.2 configures FTP's home directory as `/var/ftp`, but this can be changed to `/home/ftp`. Transaction logs are stored to `/var/log/xferlog`, but this is also configurable.
syslog server	`/var/log`	A syslog server will store lots of log files in `/var/log`. Create more `/var` space, or make `/var/log` it's own partition (perhaps on it's own disk). Or syslog output could be changed to write to a different directory (perhaps underneath `/home`).
login server	`/tmp, /var/tmp`	These directories are writeable by any user, and could become completely filled.
mail host	`/var/spool`	A mail server will store incoming mail messages in `/var/spool/mail`, and outgoing message in `/var/spool/mqueue`.
news server	`/var/spool/news`	An NNTP server requires lots of space for the newsfeeds.
samba server	`/var/log/samba`	A samba server will write logs in to the `/var/log/samba` directory, but this is configurable.
web server	`/var/log/httpd`	A web server will store log files in `/var/log/httpd`, but this can be configured.

Determining what slicing strategy works best is a combination of art and science. It may take several iterations to figure out just how the partitions should best be sliced and sized for the servers' purpose. For initial development, on a single drive system, specify just a root and swap partition, install the OS, install/remove packages, and observe the result. Then reinstall, resizing and/or creating partitions to adjust. On a multi-drive system, start by specifying root and swap partitions for the first drive, and the application (or home) space on the second drive.

With larger and larger disk drives today, one can afford to be generous when defining slice sizes: doing so can mean fewer iterations before discovering a slice strategy that works.

2.2.2.3 Example Initial Partitioning Schemes

The following example initial partition schemes are based upon a system with 512 MB RAM and two 12 GB hard drives. These can be used as a starting point for the specific server purpose rather than the generalized scheme above. Given the smaller memory capacity, the 2x rule-of-thumb was applied to determine the swap size, and all other partitions are extremely generous in their capacities.

FTP Server

partition	size	drive	
/boot	32 MB	✓ first	
/	1 GB	✓ first	
swap	1 GB	✓ first	
/usr	2 GB	✓ first	
/var	2 GB	✓ first	
/var/log	4 GB	✓ first	
/tmp	1 GB	✓ first	
/home	1 GB	✓ first	
/home/ftp	12 GB		✓ second

Syslog Server

partition	size	drive	
/boot	32 MB	✓ first	
/	1 GB	✓ first	
swap	1 GB	✓ first	
/usr	2 GB	✓ first	
/var	6 GB	✓ first	
/var/log	12 GB		✓ second
/tmp	1 GB	✓ first	
/home	1 GB	✓ first	

File Server

partition	size	drive	
/boot	32 MB	✓ first	
/	1 GB	✓ first	
swap	1 GB	✓ first	
/usr	2 GB	✓ first	
/var	2 GB	✓ first	
/var/log	5 GB	✓ first	
/tmp	1 GB	✓ first	
/home	12 GB		✓ second

Mail Server, News Server

partition	size	drive	
/boot	32 MB	✓ first	
/	1 GB	✓ first	
swap	1 GB	✓ first	
/usr	2 GB	✓ first	
/var	2 GB	✓ first	
/var/log	4 GB	✓ first	
/var/spool	12 GB		✓ second
/tmp	1 GB	✓ first	
/home	1 GB	✓ first	

Web Server

partition	size	drive	
/boot	32 MB	✓ first	
/	1 GB	✓ first	
swap	1 GB	✓ first	
/usr	2 GB	✓ first	
/var	2 GB	✓ first	
/var/log	4 GB	✓ first	
/tmp	1 GB	✓ first	
/home	1 GB	✓ first	
/home/www	12 GB		✓ second

2.2.3 Boot Loader

c) _____ select GRUB boot loader

[✓] select the GRUB boot loader.

[✓] select install on Master Boot Loader.

2.2.4 Set GRUB Password

d) _____ set GRUB password

Setting a GRUB password provides additional console security. This requires knowing the GRUB password to pass arbitrary commands to the kernel during boot time.

2.2.5 Configure Network Interfaces

e) _____ configure network interfaces

Configure the network interface/s: host name, IP address/es, default router, DNS server/s.

2.2.6 Firewall Rules

f) _____ select no firewall

At this time, select no firewall; packet filtering firewall rules will be set later.

2.2.7 Initial Root Password

g) _____ set initial root password at installation time

Maintain good password management from the outset, before the system goes live on the network. Set the root password during initial installation, as this may be forgotten later.

Do not define any accounts at this time! Define accounts later after password defaults (aging, minimum length) have been configured.

2.2.8 Authentication Configuration

h) _____ enable shadow passwords and MD5 passwords

[✓] Enable Shadow Passwords

[✓] Enable MD5 Passwords

Without the shadow file, encoded passwords would be viewable to a user. MD5 should be used to encode passwords instead of DES, as the MD5-based password encoding will take longer to crack.

2.2.9 Initial Package Group Selection

With the custom installation, packages can be selected by group. Below is a listing of the Red Hat Linux 7.2 package groupings, and an indication if that package group should be selected. If a package group is not selected, it is because it is not required for the server. Some of the default configurations delivered with these packages are contrary to good security principles, but will be addressed later on in this document. Those package groups marked with a checkmark [✓] are necessary for the implementation detailed in this document.

i) _____ select packages

Printing Support
Classic X Window System
X Window System
Laptop Support
GNOME
KDE
Sound and Multimedia Support

[✓] Network Support – include

Dialup Support

Messaging and Web Tools
Graphics and Image Manipulation
News Server
NFS File Server – *include if NFS server*
Windows File Server – *include if Samba server*
Anonymous FTP Server – *include if anonymous FTP server*
SQL Database Server
Web Server – *include if web server*

[✓] Router / Firewall – include
DNS Name Sever

[✓] Network Managed Workstation – include
Authoring and Publishing
Emacs

[✓] Utilities – include
Legacy Application Support
Software Development

[✓] Kernel Development – include
Windows Compatibility / Interoperability
Games and Entertainment
Everything

2.2.10 Create Boot Disk

j) _____ create boot disk

After applying the packages to the system, one can create a boot disk for use during situations where the system has troubles booting from the hard disk. However, with the production BIOS configuration set to disallow booting from removable media, practical application of the boot disk is limited to just the development environment.

As described later in the "System Backup" section (§4), methods other than a boot floppy can be utilized in emergency situations.

2.3 Post-Installation Package Management

2.3.1 Package Adds

After the initial OS installation completes, there are a few additional packages that are good to add (found in the `RedHat/RPMS` directory on the CD listed below).

a) _____ apply additional packages

To Add Packages:

Insert the Red Hat Installation CD into the system.

```
# mount -t iso9660 -o ro /dev/cdrom /mnt/cdrom
# cd /mnt/cdrom/RedHat/RPMS
# rpm -i package
# cd /
# umount /mnt/cdrom
# eject cdrom
```

Repeat for packages on second disk.

	package	disk	description
`rpm -i`	`libcap*`	1	libcap, required for ntp
`rpm -i`	`ntp*`	1	network time protocol

rpm –i	pdksh*	2	korn shell
rpm –i	zsh*	2	Z shell
rpm –i	lslk*	2	list local locks
rpm –i	ethtool*	2	Ethernet configuration tool

2.3.2 Package Deletes

b) _____ remove unneeded packages

Consider removing these packages:

Package Updating Utilities

rpm –e	up2date-gnome up2date rhn_register-gnome rhn_register	*To maintain a known state of the system and for production change control management, do not subscribe to Red Hat's automatic package updates (Red Hat Network).* *Why? If retained, the system could, without your being fully aware or having tested it first, introduce an updated package, possibly altering a configuration you intentionally installed..*

Web Server

rpm –e	tux webalizer apache-devel mod_dav mod_perl mod_ssl asp2php apache	*If not a web server, delete these packages.* *Why? Service minimalization, less avenues for security attacks to be applied to a system.*

Web Proxy

rpm –e	squid	*If not a web proxy server, delete.* *Why? Service minimalization, less avenues for security attacks to be applied to a system.*

FTP Server

rpm –e	anonftp	*If not an anonymous FTP server.* *Why? Package delivers the configuration for anonymous FTP.*
rpm –e	wu-ftpd	*If not an FTP server.* *Why? Service minimalization. Package delivers the utilities providing FTP capability.*

NIS

rpm –e	yp-tools ypbind ypserv	*If not an NIS server or client, delete.* *Why? Service minimalization. Eliminates possibilities of system listening to false NIS server.*

DNS Client

rpm –e	bind-utils	*If not a DNS client, delete.* *Why? Service minimalization. Eliminates possibilities of DNS spoofing and other host-name switching attacks.*

DHCP Client

rpm –e	dhcpcd	*If not a DHCP client, delete.* *Why? Service minimalization.*

System Services

rpm –e	apmd	*Laptop automatic power management. Service minimalization.*
rpm –e	talk-server talk	*Not needed for a server without users.*
rpm –e	finger-server finger	*Utilities reveal system/user information. Utilities not*

	rusers-server rusers rwall-server rwho	*needed for a system without users.*

NFS

rpm –e	nfs-utils portmapper	*If not an NFS server, delete these packages. Service minimalization.*

Samba

rpm –e	samba samba-client samba-common	*if not a Samba server, delete these packages Service minimalization.*

System Minimalization: Games, Multimedia, Console Applications, GUI, e-mail

Utilities not needed for a system without users.

rpm –e	cdda2wav cdp cdlabelgen cdparanoi cdrecord-devel cdrecord	*CD utilities*
rpm –e	gnome-games-devel gnome-games	*games*
rpm –e	arts audiofile-devel audiofile aumix gsm-devel mpg321 sndconfig sox	*sound utilities*
rpm –e	desktop-backgrounds	*GUI backgrounds*
rpm –e	gnome-user-docs	
rpm –e	gaim micq	*instant messenger utilities*
rpm –e	abiword dia ee galleon ImageMagick koffice Mesa-devel mozilla-mail mozilla-psm	*GUI / console applications*
rpm –e	balsa fetchmail metamail pine elm	*mail utilities*
rpm –e	bug-buddy	

Compliers and Development Tools

rpm –e	bison byacc flex lclint gdb gcc-g77	*Application development and languages compiler packages. Why? Having these on a system allows an attacker who has broken into the system to compile Trojan Horses directly on the system.*

2.3.3 Packages to Update

All packages should be updated with the latest release from Red Hat. The Red Hat Network allows for automated updates of the installed system; in a change-controlled environment, these automated updates can induce change (and possible error) without knowing about it. In order to maintain a closed, consistent configuration, package updates will need to be researched individually.

Since a new system is being prepared right now, start with the latest security patches and fixes. The packages listed below should expressly be updated to the latest release, because at the time of this writing, the package versions included on the Red Hat Linux 7.2 media is known to have one or more security vulnerability issues included in it.

For best implementation of package updates, updates should be downloaded and burned onto a CD, and that CD inserted into the build system; the build system should not be connected to a live network until the packages have been updated and all the below hardening configurations completed.

2.3.3.1 Getting the Latest Packages

To get the latest packages, ftp to the Red Hat updates site and download the required packages. Mirror sites are listed at http://www.redhat.com/download/mirror.html. Contents for the packages utilized in this document are located in three distribution directories: i386, i686, and noarch.

```
# ftp updates.redhat.com
```

```
ftp> bin
ftp> cd 7.2/en/os/i386
ftp> get <package-name>.rpm
ftp> cd ../i686
ftp> get <package-name>.rpm
ftp> cd ../noarch
ftp> get <package-name>.rpm
ftp> quit
```

Install the Package Updates

Use the RPM utility to update the packages:

```
# rpm -Fvh <package-name>.rpm
```

`-F` to install update only if the package already exists (e.g. don't install anything new). This is the Red Hat recommended option.
`-v` for verbose output.
`-h` for hash output to show installation progress.

```
# rpm -Uvh <package-name>.rpm
```

`-U` to update a package: this will install the package even if it didn't exist previously.
`-v` for verbose output.
`-h` for hash output to show installation progress.

```
# rpm -i <package-name>.rpm
```

`-i` to install the kernel update (one can have more than one kernel version installed). Without this `-i` option, the existing kernel source package will be overwritten, which may not be preferred.

Packages With Security-related Updates

at
cyrus-sasl, cyrus-sasl-plain, cyrus-sasl-md5
glibc, glibc-common
iptables
kernel, (or kernel-sml), modlib
openldap, openldap-clients
openssh, openssh-askpass, openssh-clients, openssh-server
stunnel
sudo
util-linux

2.4 Hardening the System

This section lists system configurations that should be standard across all servers, no matter the functional purpose of the system.

2.4.1 Configure Password Security

The user password is in the forefront of system security. Many exploits require login access to the system, and having strong account password security can reduce this risk. Password security is supported by the selection of the password encryption method, employing shadow passwords, a long minimum password length, routine password changes, and strong password checking on a proposed new password.

a) _____ enable md5 hashing, passwords stored in `/etc/shadow`

These options should be selected during the Linux installation.
Without the shadow file, encoded passwords would be viewable to a user. MD5 should be used to encode passwords instead of DES, as the MD5-based password encoding will take longer to crack.

It is possible for an attacker to pre-compute many DES encoded passwords, and if the password list is acquired through some means, to run a simple comparison between the encoded password and the pre-computed list.

TEST: To verify these capabilities are enabled:

1. check for the rpm "shadow-utils". This package must exist.
   `# rpm -qa shadow-utils`
2. inspect the `/etc/passwd` file. The second field must be an "x".
3. inspect the `/etc/shadow` file. The second field for an account with a password must be a jumble of characters 34 characters in length. If it is 14 characters in length, then it is a DES password.

> *A Side Note on Policy:*
> Some of the configuration settings suggested here – such as the password expiration cycle of 90 days, or minimum password length of 10, set in steps b) and c) below – are values which need to be carefully considered before implementation. The values suggested here are just that: suggestions. These settings would support stronger security, but may be too severe to be acceptable. The final settings should be reviewed and approved internally before implementation.

b) _____ set password expiration for users

Require a user to change passwords every 90 days or sooner. Adjust the value '90' as appropriate per local security policy. Standard Red Hat Linux 7.2 sets the default password expiration to 99999 days (280 years – effectively never).

→ edit `/etc/login.defs`, change entry:

```
PASS_MAX_DAYS                   90
```

TEST: with a user account, change the password-last-changed field in `/etc/shadow`, and attempt to log in.

`# vi /etc/shadow`

edit the password-last-changed date for the user, this is the third field (11563 in the example):

```
user:encrypted-password:11563:0:90:7:::
```

and set the last-changed date back 100 days (subtract 100 from the number 11563).

Next, try to log in as that user:

```
$ telnet localhost
login: user
Password: enter-correct-password
You are required to change your password immediately (password aged)
Changing password for user
(current) UNIX password:
```

If this "password aged" entry is not observed, the account may not currently have password aging enabled. This is possible if the account was created prior to instituting the password aging date in `/etc/login.defs`. To remedy previously-created accounts, manually apply the password aging interval (90) into the 5th colon-separated field in `/etc/shadow`.

c) _____ set minimum password length

Require users to have a long password length. Adjust the value '10' as appropriate per local security policy. Standard Red Hat Linux 7.2 installation requires just five characters for a password, which is too short for a good password.

→ edit `/etc/login.defs`, change entry:

```
PASS_MIN_LEN                    10
```

TEST: under a user account, attempt to set the password to something fewer than 10 characters.

```
$ passwd
Changing password for user
(current) UNIX password: <current-password>
New password: foo
BAD PASSWORD: it's WAY too short
New password: <CTRL-D>
```

> *On Password Length:*
> Traditionally, only the first 8 characters of a UNIX password were significant. A password could be longer, but anything after the first eight are truncated. So, "HelloDolly1" was effectively the same password as "HelloDolly7". But Linux removes this 8 character limitation. For compatibility with other UNIX systems, one may wish to retain the 8 character limit tradition. However, with faster and faster CPUs these days, the shorter a password, the more likely some password guessing or cracking program might be able to break the password. A longer password would provide additional safeguards against password guessing, in the event that the encrypted passwords are somehow acquired.

d) _____ set failed password delay

This sets the delay a user must wait after entering the wrong password. Increasing this value will frustrate brute-force password guessing. Adjust the value '5' as appropriate. Standard Red Hat Linux 7.2 media does not set this in this file, but defaults to a 3 second delay.

→ edit `/etc/login.defs`, add entry:

```
FAIL_DELAY                5
```

e) _____ enable logging of all unknown failed logins

Log all unknown failed logins.

→ edit `/etc/login.defs`, add entry:

```
LOG_UNKFAIL_ENAB          yes
```

f) _____ verify strong password checking before accepting updated password

With this setting, the guessability-strength of a proposed password can be checked. This PAM module utilizes cracklib, based off the password guessing program 'crack'. Without this module setting, a user can enter a dictionary-based password that's easily guessable by the program 'crack'. This is installed by default, but verify that it is configured by checking the authorization PAM configuration file `/etc/pam.d/system-auth`:

```
password    required    /lib/security/pam_cracklib.so retry=3 type=
```

TEST: under a user account, attempt to set the password to a dictionary word.

```
$ passwd
Changing password for user
(current) UNIX password: <current-password>
New password: arizona
BAD PASSWORD: it is based on a dictionary word
New password: <CTRL-D>
```

2.4.2 Configure Console Security

Protect the system from someone commandeering root while at the system console.

a) _____ disable console ctrl+alt+del reboots

Disable ability to rebooting the system by pressing Ctrl-Alt-Del key combination at the console.

→ edit `/etc/inittab`, change the line:

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

to:

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

TEST: the system must be rebooted for this configuration change to take effect.
To test, press <ALT-CTRL-DEL> combination, and no system shutdown should be initiated.

This configuration provides additional console security – someone with console access needs to know the root password to conduct a graceful shutdown. Perhaps more frequent is it also prevents one from accidentally rebooting the system, as this same keystroke is used with great frequency on other operating systems and easily becomes a habit.

b) _____ require root password for single-user mode login

Require root password to boot to single-user mode during reboot. Without this, someone with console access could power-cycle the machine, boot into single user mode, and take complete control of the system. Single user mode is root access to the system, from which one could introduce any number of compromises.

→ edit `/etc/inittab`, immediately following the existing line:

```
si::sysinit…
```

add the line:

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

c) _____ disable remote root logins, require remote logins be regular users instead of root

Force a system administrator to log in with a user account and then use su or sudo to acquire root privileges. With the user login and subsequent su/sudo, there is traceability of who has root privileges; without it, someone can directly log in as root and after the fact it may be impossible to properly identify who initiated that root access connection.

→ verify that `/etc/securetty` includes entries for only:
`vc/1` to `vc/11`, and `tty1` to `tty11`, inclusive.
These are the tty's which root will be allowed to directly log in at.

TEST: try to log in remotely as root.

```
$ telnet localhost
login: root
Password: <enter the correct root password>
```

No clear indication will be given that remote root logins are not allowed; this is indistinguishable from having entered an incorrect password.

2.4.3 Faceless Account Maintenance

A faceless account is an account that exists on the system yet has no real person owning it. Faceless accounts are those such as "`bin`", "`nobody`", and "`daemon`". No real person should log into these accounts; these accounts exist for individual system processes to run at a non-root level and to identify process and file ownership.

2.4.3.1 Delete Faceless Accounts?

To delete faceless accounts, or not to delete faceless accounts? Either point can be argued. Here are some points to consider:

On the one hand, the reasons for the faceless account is to have processes run at non-root permissions and for a method of associating a specific file or process to a specific function. If a file or process is owned by "`apache`", it's reasonable to assume it has something to do with the web server. If the process wasn't running as "`apache`", it would otherwise be running as "`root`" or as "`nobody`".

There is a good degree of risk running the httpd process as "root", in case some new exploit is uncovered. There are some situations where "nobody" also has special meaning, which has the potential for being exploited in some manner. Instead, a standard non-privileged account is made.

On the other hand, simply having that account listed in /etc/passwd and /etc/shadow, makes it an avenue for compromising the server. While the account is not used for logins, if the account were somehow broken into, there is not a normal user to notice that the password has changed or that the "last login" timestamp is odd. These faceless accounts are protected by the measures detailed above – disabled passwords, disabled shells, and no ftp-capability – but the account still exists.

Given that certain utilities are programmed to rely upon the existence of certain faceless accounts ("apache" for the Apache web server, "ntp" for NTP services, etc.), to remove these faceless accounts from of the system and still have the system function as expected would require and extensive effort. The application may be reworked to remove the reliance upon the faceless account, but this causes a problem for future package updates – this same faceless account removal and application cleansing would need to be reapplied each time the package is updated.

2.4.3.2 Delete Superfluous Faceless Accounts

Due to this reliance upon the faceless accounts and the need to repeat the application cleansing process after each update, the balancing position is to retain the faceless accounts that are shown to own necessary files and processes, removing the others. Those remaining accounts will be locked down and their shell set to an invalid shell.

a) _____ disable faceless account logins

Edit /etc/shadow directly, and validate the second field for all accounts (except for root) contains the text "!!" or "*". Either value will "lock" the account, meaning the account has no valid password and cannot be logged into. The following will display all accounts without locked passwords (where the lock is "!!" or "*" – but note that account passwords can be locked by many ways – these two are the means which Red Hat Linux 7.2 delivers).

```
# cat /etc/shadow | awk -F: '{if ($2!="!!") if ($2!="*") print $0}'
```

b) _____ disable faceless account shells

But even locking the account with the password can be incomplete. There have been some implementations of the r* commands in which a .rhosts file entry would grant access to the account even if the account has a locked password. To guard against this, also verify the account has a non-functional shell listed.

Edit /etc/passwd directory, and validate that the last field for all accounts (except for root) contain an invalid shell such as /bin/false or /sbin/nologin.

```
# cat /etc/passwd | awk -F: '{print $1, $7}' | \
      grep -v nologin | grep -v false
```

Some accounts will have will have shells other than /bin/false or /sbin/nologin:

sync	/bin/sync	sync the filesystems
shutdown	/sbin/shutdown	shutdown the system
halt	/sbin/halt	halt the system

These accounts could be used in a denial of service scenario if the account somehow had a legitimate password set in it. For example, someone could shutdown the system without even having to log into an account first.

If using /sbin/nologin, the contents of /etc/nologin.txt can be customized to display a message. Refer to the man page for nologin(8) for further detail.

Before considering any faceless account for deletion:
1) inspect the filesystem for any files owned by the account in question:
```
# find / -user user
```
and
2) check the process table for any processes owned by the account:
```
# ps –fade | grep user
```

To delete the user, issue the command:
```
# userdel user
```

On a standard Red Hat Linux 7.2 installation, other faceless accounts and their default shells can exist:

rpm	/bin/bash
news	-no-shell-
mailnull	/dev/null

These accounts should be considered for deletion, or minimally have their shells set to `/bin/false` or something similar.

2.4.3.3 A Faceless Account User Shell Tracker

An additional measure to protect a faceless account would be to create a login shell whose sole purpose is to send an alert in the event that the account is logged into. This alert could be sent by syslog, mail, or other mechanism. When this utility is inserted as a shell of a faceless account, in the event of a locked password compromise or `.rhosts` compromise, the utility will alert to the compromise. Because this method can be designed to give an immediate alert, this affords a much stronger position than to monitoring logs files for account activity after the fact.

The Appendix includes a listing for a skeleton faceless account shell utilizing syslog messaging. This utility should be compiled, statically linked, stripped of object symbols, and installed in either `/sbin` or `/etc` directory. This program could be extended to display the contents of a file, mimicking the behavior of `/sbin/nologin`.

```
# gcc –static noshell.c –o noshell
# strip noshell
# cp noshell /sbin/noshell
# chown root:root /sbin/noshell
# chmod 0555 /sbin/noshell
```

With the settings in the provided code, when account with this utility as its shell is logged into, with the syslog configuration defined later (section 2.4.6), a syslog message warning of the account intrusion will be written to `/var/log/secure`, `/var/log/syslog` (after syslog is configured per subsequent instruction) and forwarded to the central logger `loghost`.

TEST: run the `/sbin/noshell` executable from the command line, and observe that an entry is written to `/var/log/secure`.

```
$ /sbin/noshell
access denied.$          (the $ is the prompt, not part of the noshell output)
# tail /var/log/secure
date host noshell: intruder alert, noshell account 'user' accessed.
```

c) _____ disable faceless account ftp capability *(for systems with ftp)*

Standard Red Hat Linux 7.2 delivers no FTP ability for all uids below 100. This setting should agree with the setting `UID_MIN` in `/etc/login.defs`.

→ determine `UID_MIN`

```
# grep UID_MIN /etc/login.defs
```

→ edit `/etc/ftpaccess`, change entry of

```
deny-uid %-99 %65534-
```

edit the "99" in the line to the value of `UID_MIN` less 1:

```
deny-uid %-newvalue %65534-
```

Standard Red Hat Linux 7.2 gives `UID_MIN` of 500; unless this `UID_MIN` value is changed, 499 should be used in `ftpaccess`.

2.4.4 Configure sudo

The utility `sudo` allows a permitted user to execute a command as root or another user, as specified in a configuration file. With knowing the root password, it's an all-or-nothing proposition: someone who has to, for instance, perform the backups and has been granted root access to do so, that same someone can also create new accounts. Perhaps the system has an unstable application running on it that periodically needs to be restarted; permission to execute the run-control start/stop script can be delegated to the weekend and nighttime staff, and that permission limited to strictly the troublesome application. `sudo` can be used to grant superuser privileges for a pre-determined selection of commands to a pre-determined list of users; `sudo` can be configured to mete out superuser privileges without giving out the root password.

In the configuration provided with standard Red Hat Linux 7.2, `sudo` will ask for a password before issuing the command. This is the users password, not root's password. `sudo` can be additionally configured to log `sudo` usage to syslog: successful users, unsuccessful attempts, and errors. In this manner, an audit trail of superuser activities exists, whereas using the `su` command to root does not provide a per-command audit trail. `sudo` can also be configured to accept the command but not require the user to enter their password; this is a dangerous configuration in the event that a sudo-configured user walks away from a session and leaves themselves logged in.

Access to `sudo` is controlled by the configuration file `/etc/sudoers`. Edit this file with the `visudo` command, providing a safe and clean means of editing the `/etc/sudoers`, locking it so only one edit occurs at a time and syntax checking the proposed `sudoers` file before writing it.

2.4.4.1 The `sudoers` Configuration

This description provides an introduction to defining the `sudoers` access privileges.

The `sudoers` access file is structured as three major sections: alias definitions, user privilege specifications, and `sudo` default settings.

Aliases

There are four kinds of aliases: the Host_Alias, User_Alias, Runas_Alias, and Cmnd_Alias. Each alias definition is of the form:

`Alias_type` *`NAME = item1`*`[,`*`item2`*`, …]`

where *`Alias_type`* is one of the four alias kinds; *`NAME`* is a variable name composed of letters, numbers and/or underscore characters, and the first character is an upper-case letter; multiple items can be listed with an alias by separating the items by commas.

Host Aliases

```
# Host alias specification
Host_Alias        LOCALHOST=127.0.0.1
```

The Host_Alias lists the hosts which actions can be permitted or denied from. The host listing can be defined by hostname/s, IP address/es, network number/s, and/or netgroup/s (prefixed with a '+').

User Aliases

```
# User alias specification

User_Alias        FULLROOT=usera,userb
```

The User_Alias can be used to define different groups of privileged users, to be used later in the user privilege specifications. The users can be specified by name/s, user id/s (prefixed with a '#'), group/s (prefixed with a '%') and/or netgroup/s (prefixed with a '+').

Runas Aliases

```
# Runas alias specification

Runas_Alias       ALIAS=usera,userb
```

The Runas_Alias can be used to define what user a command should be run as. If this is not specified, root is the default. The Runas alias list can include the same elements as the User alias list.

Command Aliases

```
# Cmnd alias specification

Cmnd_Alias  KILL=/bin/kill
Cmnd_Alias  REBOOT=/usr/bin/halt,/usr/bin/reboot,/sbin/shutdown
Cmnd_Alias  USERMOD=/usr/sbin/usermod,/usr/sbin/userdel,/usr/sbin/vipw
Cmnd_Alias  GROUPMOD=/usr/sbin/groupmod,/usr/sbin/vigr
```

The command aliases can be used to specify full pathnames to permitted commands, to be used later in the user privilege specifications. These command aliases can also be used to group multiple commands into a single alias group for simplifying user privilege specifications. Listing just the filename, as shown above, allows the user to run the command with any arguments desired. Further command argument protection can be defined: two adjacent quotes ("") indicate the command may only be run without any arguments, or if command line arguments are specified then the arguments must match exactly (or by wildcard) those given by the user issuing the `sudo` command.

User Privilege Specifications

```
# User privilege specification

userspec          hostspec=(runas) commandspec

root              ALL=(ALL) ALL
%wheel            ALL=(ALL) ALL

FULLROOT          ALL=NOPASSWD:ALL
```

The user privilege specification grants privileges to specific users. Privilege can be restricted to a specific host and/or commands, and utilize the previously defined User, Runas, and Cmnd aliases.

`ALL` is a built-in alias that always causes a match to succeed. In the above specification, the root user and users in the group wheel have access to all commands.

`NOPASSWD` is a keyword that means those users will NOT have to enter their user password to complete the `sudo` operation. By default, `sudo` requires the user to enter their user password to issue a `sudo` command. In the "`FULLROOT`" example above, if a user is listed in the User_Alias `FULLROOT`, that user will not have to enter their user password to issue a command. *Use this* `NOPASSWD` *capability with care!*

Overriding Default Options

The sudoers file can also be used to change many of the run-time default options defined at compile-time. The sudoers man page covers this in great depth. Here are some options to consider:

```
Defaults      logfile=/var/log/sudo.log, log_year, log_host
```

Log `sudo` events to specified log file, in addition to syslog. With `log_year`, log the four-digit year to the `logfile`. With `log_host`, also include the hostname in the `logfile`.

```
Defaults      verifypw=all
```

The verify-password option controls if a password will be required when a user runs `sudo` with the `-v` option. Possible `verifypw` values are:

`all`	User must enter password, unless the user has a definition of NOPASSWD for all of that users' entries for the current host (default).
`any`	User must enter a password, unless the user has a definition of NOPASSWD for at least one of that users' entries for the current host.
`never`	User never needs to enter a password to use the `B<-v>` flag.
`always`	User must enter a password to use the `B<-v>` flag.

```
Defaults      ignore_dot
```

Ignore '.' and '' (current directory) in `$PATH`; `$PATH` itself is not modified. Good additional security measure in the event sudoers is misconfigured without a fully-qualified path to a command.

```
Defaults      path_info=SUDO-PATH
```

`sudo` will normally use the user's `$PATH`, but that can be over-ridden. As with `ignore_dot`, this is a good security-in-depth defense measure.

TEST:

1) configure user access to the 'kill' command. With `visudo`, make the following entries:

```
User_Alias  FULLROOT=user
Cmnd_Alias  KILL=/bin/kill
FULLROOT    ALL=NOPASSWD:ALL
```

2) initiate 'sleep' process as root
   `# sleep 600`
3) as a user in another session, attempt to kill that process through sudo
   `$ ps -fade | grep sleep`
   `$ sudo kill <pid-of-sleep>`
   `Password: <enter-user-password>`
   and the sleep process should be terminated.

2.4.4.2 Why Use Sudo At All?

Use of `sudo` is recommended, even for users who do have the root password and full rights to the system. By using `sudo`, there is an audit trail of each command issued, and, in the event that ones session is being monitored (electronically, or by someone peering over shoulders), the root password never actually gets typed.

Having to use `sudo` to obtain superuser privileges may be considered an inconvenience, but root is a specialized account that the system inherently trusts. Root has access to everything, and it is trivial for one to completely trash the system with an inadvertent keystroke. Aside from protecting oneself, the system administrator, it forces one to use the system as a user. The system should be usable by users; if one uses the system as a user and encounters difficulty, the users will too. Using a system as a user can become an early warning system for system problems, configuration errors and the like.

2.4.5 Limit `su`

The utility `su` can be restricted to assist with forcing the use of `sudo`. The ability to `su` to root can be configured with PAM to require the user to be in the 'wheel' group. Doing this requires a user's account to be previously configured to permit using `su`. Without this configuration, someone who happens to learn the root password who is not supposed to have root capabilities, cannot `su` to root without first being configured to `su` by inclusion in the wheel group. Adding this configuration

introduces a chicken-and-egg scenario that will frustrate and effectively block rogue `su`-ing. (However one can still log in as root on the console, so console physical security is necessary too). Note that this wheel group setting only applies to `su`'s to root, not `su`'s to other accounts.

a) _____ limit su ability to wheel group

As per the instructions in the comment fields of this file.

→ 1) edit `/etc/pam.d/su`, commenting out the line:

```
#auth       sufficient    /lib/security/pam_rootok.so
```

→ 2) uncommenting the line:

```
auth        required      /lib/security/pam_wheel.so use_uid
```

CAUTION: there are two entries that look similar to configuration in step 2 above: one with "trust" and one without. If "trust" is included, then a user who is in the wheel group does NOT have to enter the root password to '`su`' to root – e.g., that user is trusted. For stronger security, require the user to know the root password to '`su`' to root, and *DO NOT* include the "trust" entry.

TEST: 1) After making the above `/etc/pam.d/su` edits, attempt to `su` to root from a user not in the wheel group. It will appear as though the entered password is incorrect.

```
$ su
Password: <enter correct root password>
su: incorrect password
```

b) _____ add user to wheel group

To grant a user the ability to use `su`, add them to the wheel group by:

```
# usermod –G10 username
```

or by directly editing `/etc/group`.

TEST: 2) After adding the user to the wheel group, again attempt to `su` to root; this su should be successful.

2.4.6 Configure Logging

The kernel, daemons and applications generate messages that the syslog daemon can record for later auditing. Messaging can also be used to provide real-time alerts about such conditions as full disk partitions, CPU-bound processes, missing processes, and other performance criteria. It is important that one monitors the system log files, but in order to do so the logs must be collected first. Here we will be concerned with just getting the log information collected – what to look for in the log files is beyond the scope of this document.

To maintain log file integrity, log entries should be stored both locally on each server and on a remote central log server. In the configuration described below in Step c), there is a site-based log server (`loghost`) that receives messages from each server within site.

The file `/etc/syslog.conf` configures where the log messages will be routed. A message is classified by priority, which is composed by the facility and the level. A listing of facility and level codes can be found in `/usr/include/sys/syslog.h` and are listed here.

syslog Priority Names

Priority Names	Priority	Description
emerg, panic*	0	system is unusable.
alert	1	action must be taken immediately.
crit	2	critical conditions.
err, error*	3	error conditions.
warning, warn*	4	warning conditions.
notice	5	normal, but significant condition.
info	6	informational message.
debug	7	debug message.

* panic, error, warn are deprecated priority names maintained for backwards compatibility; these should not be used with new utilities and applications.

syslog Facility Codes

Priority Names	Description
auth, security*	security and authorization messages
authpriv	security and authorization messages (private)
cron	cron, at
daemon	system daemons
ftp	ftp daemons
kern	kernel messages
lpr	printer subsystem messages
mail	mail subsystem messages
news	news subsystem messages
syslog	syslog messages
user	user process messages
uucp	UUCP subsystem messages
local0 – local7	locally defined

* security is a deprecated facility code maintained for backwards compatibility; this should not be used with new utilities and applications.

The syslog.conf does not have a "find first match and exit" policy; log entries will be routed and recorded in each of the classifications in patches – it may be recorded multiple times if configured as such. There is nothing necessarily incorrect if a message is logged more than once other; it can, however, mean duplicate entries in the log files.

Syslog is configured by default to save log information in the /var/log directory, but this can be changed by the filenames specified in the syslog.conf configuration.

a) _____ enable additional message logging

→ edit /etc/syslog.conf, add the following entries:
(be certain that the whitespace between the first and second columns are <tab>s, not spaces)

```
# log warnings, errors to syslog
*.warn;*.err                        /var/log/syslog

# log kernel messages to kernel log file
kern.*                              /var/log/kernel
```

b) _____ enable logging to console virtual terminals

Display system logs real-time to console virtual terminals 5 and 6 (Alt-F5 and Alt-F6).
→ add the following entries to /etc/syslog.conf

```
# log majority of messages to virtual console 5
*.info;mail.none;authpriv.none      /dev/tty5
authpriv.*                          /dev/tty5
*.warn;*.err                        /dev/tty5
```

```
kern.*                                  /dev/tty5

# log mail messages to virtual console 6
mail.*                                  /dev/tty6
```

The selection of virtual terminals 5 and 6 is somewhat arbitrary. Terminal 7 is the console GUI, if that has been enabled. Users will usually proceed to terminals 2, 3, and so on in succession, so 5 and 6 have a less likely chance of being utilized as a login session and allow for a common configuration between GUI-enabled and non-GUI systems.

c) _____ configure remote syslog logging

Configure syslog and klog (the kernel message logger) to log messages to the central logger.

→ add the following entries to `/etc/syslog.conf`

```
# forward warnings, errors, auth messages to central syslog server
*.warn;*.err                            @loghost
authpriv.*;auth.*                       @loghost
```

The IP address of "loghost" should be defined in `/etc/hosts`. In this way, this entry is not susceptible to DNS problems such as slow lookups or spoofing.

d) _____ create new log files, set permissions

→ create log files

```
# cd /var/log
# touch           syslog kernel
# chown root:root syslog kernel
# chmod 0600      syslog kernel
```

e) _____ configure log rotator for new logfiles

→ add the following script fragment to `/etc/logrotate.d/syslog`

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

f) _____ configure log rotation history retention and compression

→ in `/etc/logrotate.conf`, there are two entries consider values for:

```
# keep 4 weeks worth of backups
rotate 4
```

Change the value '4' to the number of weeks worth of logs to keep. Four weeks may be sufficient for an individual server, but for a central loghost should keep more log depth: refer to Section 3.2.5 for central loghost configuration.

```
# uncomment this if you want your log files compress
#compress
```

If the logs are compressed, the logs will consume less disk space but will have to be uncompressed each time they are viewed.

2.4.7 Configure Service Banners

Many services display welcome banners that report information very useful to those wishing to gain entry into the system. Sometimes the OS brand name and version will be displayed, other times the application version – all of which is information that does not need to be revealed. Reporting such information trivializes the task of identifying which exploits to employ. Version concealment assists individual system security – without such version knowledge, an attempted hacker will have to employ the entire smorgasbord of exploits, and doing so can slow them down and/or get them discovered; whereas if the specific version is known up-front, the attack can be more focused.

a) _____ change `/etc/issue`

`/etc/issue` is displayed during a telnet session prior to the user entering in their name and password. In earlier releases of Red Hat Linux, the run control script `S99local` would re-write the contents of `/etc/issue` during each reboot. Instead of modifying the `S99local` script, a new RC script should be created which follows `S99local` to write the contents of `/etc/issue` (if the `S99local` script were customized, a subsequent patch update could undo the customizations and leave the system back with the standard Red Hat Linux 7.2 `/etc/issue` contents). While 7.2 does not over-write `/etc/issue` in `S99local`, the method of creating an RC script to re-write `/etc/issue` is compatible with prior releases.

At the minimum, the `/etc/issue` file should *not* report the OS brand or version (e.g. "Red Hat Linux 7.2"). The message in `/etc/issue` should display an organization-approved appropriate use message. If this message displays an identical message on all UNIX systems across the environment, it will be impossible to tell from the telnet banner alone if it's a Linux system or some other (UNIX or anything else).

→ create `/etc/rc.d/init.d/zzzissue`

```
#!/bin/sh

# zzzissue
# chkconfig: 2345 99 99
# description: write contents of /etc/issue

PATH=/bin
rm -f /etc/issue

echo " > > > AUTHORIZED USE ONLY < < <
enter your disclaimer / appropriate use message here" > /etc/issue

chown root:root /etc/issue
chmod 0444 /etc/issue
cp -p /etc/issue /etc/issue.net
```

→ install the run control script

```
# chown root:root /etc/rc.d/init.d/zzzissue
# chmod 0555 /etc/rc.d/init.d/zzzissue
# chkconfig --add zzzissue
# /etc/rc.d/init.d/zzzissue
```

TEST: telnet to the system and observe. The message included in `/etc/issue` should be displayed prior to the "login:" prompt.

```
$ telnet localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

 > > > AUTHORIZED USE ONLY < < <
login:
```

b) _____ change `/etc/motd`

`/etc/motd` (message of the day) is displayed during a login session following the user entering their correct password. As there is no rc script managing `/etc/motd`, this file can be directly edited. The `/etc/motd` message should re-iterate the appropriate use message.

→ edit `/etc/motd`, enter content

If there is concern about the content of `/etc/motd` "drifting" over time, it can be reasserted by incorporating it into the `zzzissue` script above.

c) _____ change sendmail banner

As delivered from the standard Red Hat Linux 7.2 media, a telnet to the sendmail port (25) will display sendmail version information. In this step the sendmail configuration file is edited to display a simple, non-revealing message. Later sendmail will we configured to run not in daemon mode. It is still a good idea to universally clean up the sendmail banner in case sendmail services are later activated in daemon mode; at that time it will likely be forgotten to clean up the sendmail banner.

→ edit `/etc/sendmail.cf`, change the line:

```
O SmtpGreetingMessage=$j Sendmail $v/$Z; $b
```

to:

```
O SmtpGreetingMessage=sendmail ready
```

To go even further, delete the "sendmail" name altogether, giving a greeting like:

```
O SmtpGreetingMessage=ready
```

or introduce an element of misdirection by including a bogus name such as "OpenVMS Mail".

```
O SmtpGreetingMessage=OpenVMS Mail
```

TEST:

1) Restart the sendmail daemon

```
# /etc/rc.d/init.d/sendmail restart
```

2) Verify the SMTP greeting message has been changed:

```
# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 ready ESMTP     (the message between '220' and 'ESMTP' should be the as specified above)
QUIT
221 2.0.0 xyz.domain.com closing connection
Connection closed by foreign host.
```

d) _____ change ftp banner *(for systems with ftp)*

→ edit `/etc/ftpaccess`, add entry

```
greeting terse
```

which will now display the message:

```
220 FTP server ready.
```

instead of the default version-revealing message:

```
220 nodename FTP server (Version wu-versionnumber) ready.
```

TEST: Verify the FTP greeting message has been changed:

```
# ftp localhost
Connected to localhost (127.0.0.1).
220 FTP server ready.
Name (localhost:root):
```

2.4.8 Set Network Parameters for Security

There are some settings for the IP stack which can prevent the system from being susceptible (or less susceptible) to certain network probes and attacks.

a) _____ disable source-routed packets

Turn off source-routed packets to stop packets from being sent between interfaces.

→ edit `/etc/sysctl.conf`, add entry

```
# disable source-routed packets
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.all.forwarding = 0
```

b) _____ increase headroom for handling half-open sockets

Mitigate "SYN flood" attacks by increasing the number of slots available for half-open sockets.

→ edit `/etc/sysctl.conf`, add entry

```
# protection for syn floods
net.ipv4.tcp_max_syn_backlog = 1280
```

c) _____ ignore ICMP echo broadcast requests

Mitigate Smurf attacks by ignoring ICMP echo broadcast requests.

→ edit `/etc/sysctl.conf`, add entry

```
# ignore ICMP echo broadcast requests
net.ipv4.icmp_echo_ignore_broadcasts = 1
```

d) _____ enable source route verification

Block IP spoofing by enabling source route verification.

→ edit `/etc/sysctl.conf`, add entry

```
# enable source route verification: block spoofing
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1
```

e) _____ additional IP network tuning

→ edit `/etc/sysctl.conf`, add entry

```
# disable ip forwarding
net.ipv4.ip_forward = 0

# enable TCP SYN cookies protection
net.ipv4.tcp_syncookies = 1

# disable ICMP redirects
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.all.accept_redirects = 0

# enable logging for oddities
net.ipv4.conf.default.log_martians = 1
net.ipv4.conf.all.log_martians = 1

net.ipv4.icmp_ignore_bogus_error_responses = 1
```

f) _____ activate changes now

→ validate these settings by activating them now:

```
# /sbin/sysctl -p /etc/sysctl.conf
```

2.4.9 Configure Run Control Scripts

The Run Control scripts, which reside in the `/etc/rc.d/init.d` directory and are linked into the `/etc/rc.d/rc[0123456].d` directories, control which scripts get executed on system start-up, shutdown, and when changing between run-levels.

2.4.9.1 Create New RC Scripts

Several new scripts should be added to reassert proper system settings at each system reboot.

a) _____ add script to set umask

Create a run-control script to force the correct umask for all subsequent run-control scripts. Settings to consider for umask:

0022 allow world- and group-read access.
0027 turn off world-read access, enable group-read access (stronger security than 0022).
0077 turn off world- and group-read access (best, stronger security than 0027).

→ create `/etc/rc.d/init.d/umask`

```
#!/bin/sh

# umask
# chkconfig: 2345 01 01
# description: force umask

# choose best setting for your environment
#umask 0022
#umask 0027
umask 0077
```

→ install the run control script

```
# chown root:root /etc/rc.d/init.d/umask
# chmod 0555 /etc/rc.d/init.d/umask
# chkconfig --add umask
# chkconfig umask on
```

b) _____ add script to reassert correct permissions to `/tmp`, `/var/tmp`

Just in case these permissions change over time, reassert correct permissions. Only with the sticky bit o+t set on the directory will a user be able to delete files owned by that user only; without it one user can delete files created by another user.

→ create `/etc/rc.d/init.d/tmpfix`

```
#!/bin/sh

# tmpfix
# chkconfig: 2345 79 79
# description: reassert permissions on /tmp, /var/tmp

PATH=/bin

if [ -d /tmp ]; then
  chown root:root /tmp
  chmod 1777 /tmp
fi

if [ -d /var/tmp ]; then
  chown root:root /var/tmp
  chmod 1777 /var/tmp
fi
```

→ install the run control script

```
# chown root:root /etc/rc.d/init.d/tmpfix
# chmod 0555 /etc/rc.d/init.d/tmpfix
# chkconfig --add tmpfix
# chkconfig tmpfix on
# /etc/rc.d/init.d/tmpfix
```

c) _____ add script to reassert correct permissions to `/etc/passwd`, `/etc/shadow`
Guard password information carefully. Reassert correct permissions in case these change over time.

→ create `/etc/rc.d/init.d/permfix`

```
#!/bin/sh

# permfix
# chkconfig: 2345 79 79
# description: reassert permissions on /etc/passwd, /etc/shadow

PATH=/bin

chown root:root /etc/passwd /etc/shadow
chmod 0644 /etc/passwd
chmod 0400 /etc/shadow
```

→ install the run control script

```
# chown root:root /etc/rc.d/init.d/permfix
# chmod 0555 /etc/rc.d/init.d/permfix
# chkconfig --add permfix
# chkconfig permfix on
# /etc/rc.d/init.d/permfix
```

2.4.9.2 Minimize Existing Run Control Scripts

The goal is system minimalization: turn off that which is not needed. Don't know if something is really needed – try running the system without it and see if the system functions as expected. It is better to turn something off and later find out it is needed, than unknowingly leaving something enabled which can possibly be used against the system later.

To discern what is running on a system, look through the process table and open ports:

```
# ps -fade
# netstat -vatup
```

Then go through the services in the run control directories. Remove services that are not needed for the server's purpose. The list here should be considered closely. Refer to Section 5.4.2 for more detail on checking the system for running services.

There are several depths to which the RC scripts can be purged:

chkconfig off: the script in the `rc[0123456].d` directory can be removed, leaving the script in place `init.d` and easily enabled later with chkconfig.

chkconfig del: also remove the script from the chkconfig database. Doing so raises ever so slightly the hurdle for someone to later enable the service.

remove rpm: removing the RPM altogether raises the hurdle even further to restore the service.

Service	Description	Security Risk	Recommendation
gpm	mouse support for text aps	low	retain
kudzu	runs hw probe for new hardware	low	disable rc script: deployed systems shouldn't have frequent hardware changes – when hardware changes introduced, run this manually
sendmail	receive incoming mail	high	disable rc script: we will later configure sending outgoing mail
netfs	mount network file systems: NFS, SMB, NCP	medium	disable rc script, if NFS, SMB, or NCP client services not needed
lpd	print daemon	low	remove package, if printing not needed

xfs	X font server	low	disable rc script, if not providing console GUI
autofs	automount file systems on demand	low	disable rc script, if not automounting file systems
nfs	NFS server	medium	remove package, if not being an NFS server
nfslock	NFS file locking	medium	remove package, if not being an NFS server
nscd	name services cache daemon	low	remove package, if not using naming services NIS, NIS+, LDAP or hesiod
ident	provide user identity of specific tcp connection	medium	disable rc script
smb	SMB server	medium	remove package, if not being an SMB server
httpd	Apache web server	high	remove package, if not being a web server
tux	kernel-based web server	high	remove package, if not being a web server

2.4.10 Configure sendmail Service

2.4.10.1 Removing Sendmail

The FTP and syslog servers do not necessarily require sendmail to conduct their function. The best security option is to remove sendmail altogether. As sendmail frequently has security vulnerabilities uncovered, complete removal eliminates all sendmail concerns.

a) _____ remove sendmail

```
# rpm -e sendmail
```

If not certain if the systems can function without sendmail, try it without sendmail first. Sendmail can always be added back in later if it really is necessary.

2.4.10.2 Retaining Sendmail

If sendmail services is necessary, there are some initial steps which can be taken the safeguard sendmail from sendmail version and user reconnaissance. This is not intended to be a full discourse into securing sendmail, as other products, freely and commercially available, which can be used to replace sendmail.

Steps c) and c) will disable certain operations within the sendmail daemon, the later steps will disable sendmail daemon mode. Only disable sendmail daemon mode if the server will not receive emails. Even if the sendmail daemon is disabled, still disable the operations in Steps c) and c), in case the sendmail daemon mode is later established.

a) _____ get latest sendmail version

Security vulnerabilities are frequently uncovered with sendmail, resolved with frequent version updates. Refer to Section 2.3.3.1 for how to get the latest version of packages from Red Hat.

b) _____ disable HELP operation

"HELP" operation will display the sendmail version number as well. This operation can be hobbled by creating an empty help file.

→ remove sendmail help file

```
# rm /etc/mail/helpfile
```

(in earlier releases, this file was `/usr/lib/sendmail.hf`)

→ create empty sendmail help file

```
# touch /etc/mail/helpfile
# chown root:root /etc/mail/helpfile
```

```
# chmod 0444 /etc/mail/helpfile
```

c) _____ disable EXPN, VRFY operations

Standard Red Hat Linux 7.2 delivers these operations already disabled.

→ verify operations disabled, look for "novrfy" and "noexpn"

```
# grep PrivacyOptions /etc/sendmail.cf
O PrivacyOptions=authwarnings,novrfy,noexpn,restrictqrun
```

d) _____ verify HELP, EXPN, VRFY configurations

TEST:

1) Restart the sendmail daemon

```
# /etc/rc.d/init.d/sendmail restart
```

2) Verify these configurations work as intended, by telneting to port 25 and trying them:

```
# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 ready ESMTP
HELP
214 2.0.0 End of HELP info
EXPN
502 5.7.0 Sorry, we do not allow this operation
VRFY
252 2.5.2 Cannot VRFY user; try RCPT to attempt delivery (or try finger)
QUIT
221 2.0.0 xyz.mydomain.com closing connection
Connection closed by foreign host.
```

If these sendmail HELP, EXPN and VRFY commands are left enabled, sendmail could reveal system and user information. With the HELP capability, the sendmail version is displayed. Knowing the sendmail version, an attacker may know an exploit for that specific version and could apply it immediately. With the EXPN and VRFY capabilities, it is possible to conduct user reconnaissance. An 'EXPN root' would likely give the email address and/or username of a legitimate account on the system. Additionally, the username identity of the administrator should be guarded for systems like FTP and syslog: these systems normally will not have user accounts, but will likely have an account for the system administrators to log in with. Thus, 'EXPN root' will likely give an account name that will work for all systems within the site.

e) _____ set out-bound SMTP server for sendmail

Configure the smarter-relay node for sendmail for this node to forward out-bound mail to.

→ edit `/etc/sendmail.cf`, edit the following entries:

```
DSmail.mydomain.org
DRmail.mydomain.org
DHmail.mydomain.org
DMmydomain.org
```

where "mail.mydomain.org" is the fully-qualified domain name of the SMTP server.

`DS` is the "smart" relay host.

`DR` will be sent mail messages with unqualified host names (those with no "mydomain.org").

`DH` will get all local email traffic (user@node without any further domain information).

`DM` is the masquerade address.

f) _____ disable sendmail daemon mode (turn off receiving incoming emails)

Standard Red Hat Linux 7.2 configures sendmail to run in daemon mode, listening for incoming mails. To disable the receipt of in-coming messages, the sendmail daemon can be turned off.

```
# chkconfig sendmail off
```

2.4.11 Configure TCP Wrappers

TCP wrappers, installed and enabled by default with the standard Red Hat Linux 7.2, can be used to guard connectivity to TCP-based services. Protection can be defined on a per-source and per-service basis. TCP wrappers can be used with guard services managed by xinetd, and other TCP services such as portmapper (with NFS).

Access is managed by two control files: `/etc/hosts.allow`, and `/etc/hosts.deny`. The basis by which to allow service connection is evaluated based upon the configuration in these two files. The search stops when the first match is made: 1) access is granted when a (service,sourcehost) pair is matched in `hosts.allow`; 2) access is denied when a (service,sourcehost) pair is mated in `hosts.deny`; 3) otherwise access is granted. Refer to the man page for hosts_access(5) for more detail. An optional third argument can specify an action to take when there is a (service,sourcehost) match.

Given the access search rules, to prevent the "all else" rule (number 3) from taking effect, establish hosts.deny to default deny services uniformly. Later, access can be included into hosts.allow as services are needed on a case-by-case basis. Omitting this "ALL:ALL" deny rule will grant access to all services unless that service is expressly denied; including this "ALL:ALL" deny rule denies access to all services unless that service is expressly allowed.

a) _____ establish TCP wrappers to default deny

→ edit `/etc/hosts.deny`, add the following entry:

```
ALL:     ALL
```

Instead of a blanket "deny all" statement, consider a response for the specific service:

```
in.ftpd:    ALL: twist (sleep 7; /bin/echo "421 Access denied") &
in.telnetd: ALL: twist (sleep 7; /bin/echo "Access denied") &
in.rexec:   ALL: twist (sleep 7; /bin/echo "Access denied") &
in.rlogind: ALL: twist (sleep 7; /bin/echo "Access denied") &
in.rshd:    ALL: twist (sleep 7; /bin/echo "Access denied") &
in.rsync:   ALL: twist (sleep 7; /bin/echo "Access denied") &
in.fingerd: ALL: twist (sleep 7; /bin/echo "No one logged on") &
#
ALL:        ALL
```

If this method is used, be sure that the "ALL: ALL" entry occurs last; if it occurs earlier in the list, it will receive the match and be operated upon and the service-specific entries listed later in the file will not be operated upon.

In this example, the echo message is tailored for the service listed (finger gives "No one logged in", which is the standard message from the fingerd when there indeed is no one logged in). The sleep statement can be used to frustrate and slow down an attacker; to drop the connection as quickly as possible, the twist portion should be omitted from the configuration.

TEST:

For the specially configured "Access denied" services, attempt to connect to those services before granting any access with `hosts.allow`.

```
$ ftp localhost
421 Access denied
$ telnet localhost
Access denied
```

b) _____ allow services on per-service basis

→ edit `/etc/hosts.allow`, to grant access on a per-service basis:

```
sshd:    LOCAL, localhost, hosta, 10.4.5.6, 10.5., .me.com
```

Entries must exist in `/etc/hosts.allow` for each service that access it to be granted for.

This example entry grants service for incoming SSH only to the systems specified on the remainder of the line. This entry utilizes several different methods for granting per-host access:

`LOCAL`	TCP Wrappers keyword, meaning any host defined in `/etc/hosts`. Caution: this will only match the first entry in `/etc/hosts` and of that entry has no "." dot character in the name. As such, a fully-qualified domain name would not match under "LOCAL".
`localhost`	Allow access from "localhost", e.g. 127.0.0.1.
`hosta`	Allow access from the server which IP number resolves to hosta. Resolution is provided by `/etc/hosts` and/or DNS, if the system is configured as a DNS client.
`10.4.5.6`	Allow access from the system which IP address is 10.4.5.6.
`10.5.`	All access from any system in which IP address begins with 10.5.
`me.com`	Allow access from any system which name resolves to be within the '.me.com' domain.

More hostname pattern matching can be utilized, with detail in the `hosts.allow` man page.

Also, multiple services can be configured on the same line:

```
sshd, in.telnetd: LOCAL, localhost, hosta, 10.4.5.6, 10.5., .me.com
```

For an anonymous FTP server, the following hosts.allow entry would be necessary with the default deny-all shown in step a) above:

```
in.ftpd:    ALL
```

To determine what xinetd service names can be specified in `hosts.allow` and `hosts.deny`, reference the "server =" entry in the xinetd.d configuration:

```
# grep server /etc/xinetd.d/* | grep -v server_args | grep =
```

2.4.12 Configure xinetd Services

Services here should be minimalized: if something isn't needed, get rid of it. A service can be turned off by editing the individual `xinetd.d` service configuration file to include the entry "disable=yes"; stronger disabling can be achieved by out-right deleting the `xinetd.d` service configuration file for the specific service.

Services delivered by standard Red Hat Linux 7.2 installation:

Service	Description	Recommendation
chargen	character stream generator	remove, can be used in denial of service attack
chargen-udp	character stream generator	remove, can be used in denial of service attack
daytime	give time of day	remove, can be used in denial of service attack
daytime-udp	give time of day	remove, can be used in denial of service attack
echo	echo back input stream	remove, can be used in denial of service attack
echo-udp	echo back input stream	remove, can be used in denial of service attack
time	time server, used with rdate	remove, DoS vulnerability; use NTP instead
time-udp	time server, used with rdate	remove, DoS vulnerability; use NTP instead
rexec	remote execution server	remove, r* vulnerability; use SSH instead
rlogin	rlogin server	remove, r* vulnerability; use SSH instead
rsh	rcmd/rsh server	remove, r* vulnerability; use SSH instead
rsync	rsync server	remove, r* vulnerability; use SSH instead
telnet	telnet server	keep only if will allow incoming telnet service; recommend removing and using SSH instead
wu-ftpd	FTP server	keep only if will allow incoming FTP service (real user or anonymous); disable otherwise

Check the directory `/etc/xinetd.d` for other services which may have been installed if additional RPMs were added to the server. For instance, installing linuxconf package will add the service 'linuxconf-web' to this directory.

Services can also be checked with "**`chkconfig --list`**", and looking for services following the header "`xinetd` based services".

a) _____ remove unnecessary services

→ remove unnecessary services

```
# cd /etc/xinetd.d
# rm -f chargen* daytime* echo* time*
```

If all `xinetd`-based services are removed, the `xinetd` daemon can be disabled.

```
# chkconfig --del xinetd
```

Consider also removing the package altogether.

b) _____ limit access to services

Enabled services can be further tuned by limiting access within `xinetd` (in `/etc/xinetd.conf`) and/or by TCP wrappers (in `/etc/hosts.allow` and `/etc/hosts.deny`). Consider configuring both for guarding the services with two layers instead of just one.

Controlling source host access by TCP wrappers versus xinetd.conf:
TCP wrappers can see only one connection at a time; `xinetd` can be configured to limit the rate of incoming connections, the number of incoming connections from specific hosts, or the total number of connections for a service. `xinetd` can be configured to limit access to services based upon time of day, and can associate a specific service to specific addresses, providing different services based upon source (perhaps such as internal within the site versus external connections). Both methods can be used to define a specific response to a service request. TCP wrappers can be used for TCP-based services outside of xinetd.

xinetd.conf access configuration:
Enabled services can be further tuned by limiting access to said services by specific subnet. Settings can be defined for all xinetd services in the `/etc/xinetd.conf` file, or per individual service in the `/etc/xinetd.d/<service>` file. Access options are:

```
only_from = (list)
no_access = (list)
```

with (list) elements being IP addresses, IP networks, or domain names, with list elements space separated.

For instance, for a service to be available to only the localhost, specify:

```
only_from = 127.0.0.1
```

And to specify a specific subnet only:

```
only_from = 10.5.5.0/24
```

where "/24" indicates a class C address space, "/16" a class B, and "/8" a class A.

The "no_access" option can be listed to give access from everywhere except the listed location(s).

More detail on xinetd can be found at the xinetd website: http://www.xinetd.org/.

2.4.13 Configure NTP Service

Network Time Protocol keeps the system times of networks of machines synchronized. Using NTP is a very simple method to keep system times synchronized. NTP has been ported to many operating systems (UNIX and non-UNIX), and is very network bandwidth friendly. NTP should be employed to, minimally, keep audit logs consistent between servers. Other protocols and applications, such as

Kerberos or `make` (distributed software development), are time sensitive and having participating systems slightly out of synch will frustrate, even possibly break, the protocol or application.

The system will keep the best time synchronization when many NTP servers are listed instead of just a few. NTP reference servers should be both network-geographically nearby as well as far away. NTP time sources can be from publicly available NTP servers, or from locally installed GPS clocks connected up to servers within your environment. If using local GPS clocks and if your systems are located across multiple sites, distribute the clocks geographically so they will synchronize to different satellites. If distributing GPS clocks internationally, be certain to check for import/export restrictions on the GPS for the destination and source countries.

> *On Using External Public NTP servers:*
> If using publicly available NTP servers, be sure to get permission before doing so. Refer to the NTP web site http://www.eecis.udel.edu/~mills/ntp/servers.htm for further information about this. Heed this sites' suggestion of "please respect the access policy as stated by the responsible person [of the public NTP server site]". These sites are providing a courtesy service to you, so be courteous in return.

Red Hat Linux 7.2 delivers NTP version 4.1.0.

a) _____ configure `ntp.conf`

→ define the NTP server in `/etc/ntp.conf`

The following entries are minimally necessary in the `ntp.conf` file, for an environment doing directed NTP queries to an NTP server.

```
server <ntp-server-1-ip-address>
driftfile <name-of-drift-file>
```

Multiple servers can be specified on multiple lines.

The standard Red Hat Linux 7.2 provided `ntp.conf` already has the entry "`driftfile /etc/ntp/drift`".

Note that once a system is configured as an NTP client and has the NTP daemon running on it, that same system can distribute time to other systems: an NTP client can in turn be an NTP server to other NTP clients. When using a directed query-based NTP method, no additional configuration beyond what is listed above is required to turn an NTP client into an NTP server.

b) _____ verify existing system time is "close" to real time

Before starting the NTP daemon, verify that the local system time is within a few minutes of the real time. The NTP daemon will be unable to synchronize the local system time if it varies by more than 5 or 10 minutes from the time source. The utility '`ntpdate`' can be used to bring a widely varying local system clock into alignment with the source.

```
# ntpdate <ntpserver>
```

c) _____ start NTP client daemon

```
# chkconfig ntpd on
# /etc/rc.d/init.d/ntpd start
```

d) _____ verify NTP service

→ to verify NTP services, use

```
# ntpq -n -c peers
```

and check that it is connected to the NTP servers defined in `ntp.conf`, and the stratum (st) is well less than 16. If a system has a undisciplined clock (e.g., using only itself), it will have stratum 16.

```
     remote      refid      st t when poll reach   delay   offset  jitter
==============================================================================
*10.42.2.2    10.43.2.2     3 b   43   64  376    0.844    0.031   0.008
*10.42.2.3    10.44.2.3     3 b   43   64  376    0.914    0.033   0.011
*10.42.2.4    10.45.2.4     3 b   43   64  376    0.321    0.053   0.009
```

`ntpq` can also be used interactively by not specifying a command "-c *command*". There are many additional options to `ntpq`, which can be listed with the "help" option inside `ntpq` or by referring to the man page.

2.4.13.1 NTP Utilities

Other NTP utilities:

`ntpdate`	synchronize the local system clock with time from an NTP server. `ntpd` cannot be running when `ntpdate` is run.
`ntpq`, `ntpdc`	NTP query programs. `ntpq` exists with NTP v3 and v4; `ntpdc` exists with just NTP v4.
`ntptrace`	trace the chain of NTP servers back to the parent time source.

2.4.13.2 Alternative NTP Topologies

Using Network Routers as NTP Servers

An alternative to using directed queries at another system is to incorporate the network equipment. Some network routers can also run NTP, being an NTP client to keep themselves in synch, but also acting as an NTP server. As an NTP client, it can respond to direct NTP queries, and then `ntp.conf` will be as above for each system. For example, each subnet's default router, can respond to NTP queries, and, in order to maintain a common ntp.conf throughout the site, `ntp.conf` would list "server" entries for the default router for each subnet at the site. How to configure a network router as an NTP server is outside the scope of this document.

Using Broadcasting NTP

NTP servers can also be configured to provide broadcasts. For broadcasting NTP, the `ntp.conf` of the NTP client is quite different: it must specify what host/s it expects to receive broadcasts from. By specifying this, it authenticates the source. The site's network routing equipment should be configured to pass this NTP broadcast across the site, but should not pass it out onto the WAN.

Some network routers can also provide NTP broadcasts. Extending the above network router-based NTP query topology, the NTP broadcast can be generated by each subnet's default router and will be contained to just that subnet itself.

```
broadcastclient
disable auth
restrict default nomodify notrust ignore
restrict 127.0.0.1
restrict <router-IP-address>
```

line 1: broadcastclient: listen for broadcast.
line 2: disable authentication, it's on by default. However, most network routers cannot provide NTP encryption so this must be turned off.
line 3: set default to not allow remote modification of ntp behavior, trust nothing, and ignore everything.
line 4: give no restriction to localhost psuedo-clock.
line 5: and no restriction to the router's IP address.

More detail on NTP can be found at http://www.ntp.org/.
A list of public NTP servers can be found at http://www.eecis.udel.edu/~mills/ntp/servers.htm.

Configuring as a Broadcast NTP Server

To configure a system as a broadcast NTP server, it first must be configured to be an NTP client (either by listening to NTP broadcasts itself, or with direct NTP queries to other NTP servers). To become a broadcast server, add the line

```
broadcast <netmask-to-broadcast-to>
```

to `ntp.conf`. This broadcast can be limited to a specific subnet (e.g. 10.2.3.255) or to an entire site (10.2.255.255). A broadcast across multiple subnets will require network router configuration to allow such a broadcast through the networking equipment.

If systems receiving the NTP broadcast are NTP v3 (v4 is delivered with Red Hat Linux 7.2), the ntp.conf broadcast entry should be changed to

```
broadcast <netmask-to-broadcast-to> version 3
```

An NTP v4 client can receive a v3 broadcast time packet.

2.4.14 Configure SSH

Secure Shell (SSH) was originally developed to remove the problem of clear text passwords being sent over the network with `telnet`, `rlogin` and `rsh` utilities. SSH provides strong authentication using public key cryptographic algorithms and automatic, transparent encryption of network communications for login and file copy operations. SSH should be used for all interactive shell and file transfer sessions. There are SSH based tools to replace the traditional UNIX utilities `telnet`, `ftp`, `rlogin`, `rsh` and `rcp`.

Red Hat Linux 7.2 delivers Open Secure Shell version 2.9p2-7. The standard Red Hat Linux 7.2 configuration delivers SSH as a standalone daemon, uncoupled from `xinetd`; this configuration should remain. Having `sshd` standalone allows the opportunity to remove `xinetd` altogether.

a) _____ get latest version of SSH

There may be security vulnerabilities which are resolved with the latest release.
Refer to Section 2.3.3.1 for how to get the latest version of OpenSSH (delivered on the Red Hat Linux 7.2 distribution).

b) _____ enable / verify enabled SSH daemon

```
# chkconfig sshd on
# chkconfig --list sshd
```

c) _____ edit sshd_config

→ edit `/etc/ssh/sshd_config`, and add or change the following entries:

Uncomment and configure the addresses sshd should listen to:

```
ListenAddress <yournetwork>
```

(an address of 0.0.0.0 will accept connection from anywhere).

Do not allow direct root logins, change "yes" to "no":

```
PermitRootLogin no
```

Do not X11 forwarding, change "yes" to "no":

```
X11Forwarding no
```

Display our network login "issue" message, by uncommenting:

```
Banner /etc/issue.net
```

d) _____ edit TCP wrappers hosts.allow

→ edit `/etc/hosts.allow`, configure to grant access for the SSH daemon:

```
sshd:   hostslists
```

e) _____ restart SSH daemon

```
# /etc/rc.d/init.d/sshd restart
```

At this point SSH will work, and authentication depends upon the user entering their password to connect to the remote system. The session is already encrypted, so the password is not sent in clear-text as with telnet or rlogin. Authentication can also be done with host-based authentication with

public/private key pairs, Kerberos (OpenSSH supports Kerberos v4, commercial SSH supports Kerberos v5), with SecurID and PAM.

f) _____ generate SSH keys

Using a public/private key pair for SSH authentication requires the user to generate a key pair prior and copying the public key to the SSH server.

1. _____ generate key pair
   on your workstation, in your regular user account, issue the command:
   ```
   $ ssh-keygen
   ```
   to compute the public/private key pair and prompt for the save. Accept the default save file of `.ssh/identity` in your home directory.

   Next you will be asked for a passphrase. Apply good password rules when determining a passphrase. Keep in mind that this is a 'phrase', meaning it should have some length to it. SSH can be configured to not require a passphrase. This is convenient because the passphrase is not needed to open an SSH session to another system, but the downside is if there is a workstation and/or account compromise, the attacker now has an open door to all the other servers.

   After running ssh-keygen, two files will be created:

`identity`	binary file with your private key
`identity.pub`	ACSII file with your public key

2. _____ put public key on server
   The public key has to be stored on the server before SSH can be used to log in to it.
   DO NOT COPY OVER THE NETWORK!
   Doing so makes the key vulnerable to interception. Instead, the safest way to put this key onto the server is to copy it to removable media and physically transport it to the server.

   Copy the identity.pub file onto the server, and *append* it to the existing authorized_keys file; don't overwrite existing keys for other servers.
   ```
   $ cd
   $ mkdir .ssh
   $ cd .ssh
   $ cat identity.pub >> authorized_keys
   $ chmod 0600 authorized_keys
   ```
   notice the append! don't overwrite!

   Make sure the file is owned by you and is mode 0600.

   The file `authorized_keys` contains the public keys for each host you can connect from, so be certain to *append* new keys instead of overwriting.

g) _____ replace r* clients with SSH

Once SSH is installed and configured, consider replacing the r* clients (`rlogin`, `rsh` and `rcp`) with their SSH equivalents.

→ Remove or rename existing r* clients:
If these have already been removed from xinetd's configuration, the package can be removed. This forces the user of the SSH-based tools.
```
# rpm -e rsh
```
If not removing the package, consider renaming the r* clients:
```
# cd /usr/bin
# mv rlogin rlogin.old
# mv rsh rsh.old
# mv rcp rcp.old
```

→ Replace r* clients with SSH equivalents:

```
# cd /usr/bin
# ln -s slogin ./rlogin
# ln -s ssh ./rsh
# ln -s scp ./rcp
```

2.4.15 Configure Host-based Packet Firewall

A packet firewall is not just a good idea for the perimeter firewall alone. Firewall technology is a subject in and of itself and well beyond the scope of this document. Constructing a perimeter firewall is outside the scope of this document, and it is presumed this is already done; the discussion here will be limited to adding a simple packet firewall to the server as a filter for that server to assist in the protection of the systems within the internal network.

A host-based packet-filter can protect against internal threats, and also against misconfigured or inadequate perimeter firewalls.

There are several firewall utilities which can be utilized with Red Hat Linux 7.2: `ipchains` and `iptables`. Ipchains is stateless, acting only upon the information contained within the packet. Introduced with the Linux 2.4 kernel, iptables is statefull, acting upon information in the packet and history retained about previous packets – rules can take into account flow of packets.

Which method is better to use, ipchains or iptables, is subject to debate. Because ipchains works only upon the content of the individual packet, it can operate upon each packet more quickly. But because it has no memory of previous conversations, its rules are less complex and may not do everything needed. Iptables may take longer to execute, because in addition to digesting the content of the packet, it must also refer to history of previous packets, a lookup for each packet that incurs a cost. There still are some stability concerns with iptables under heavy load. For a more detailed comparison of ipchains and iptables, see the "Linux Network Administrators Guide" at the Linux Documentation Project, specifically Chapter 9 at http://www.linuxdoc.org/LDP/nag2/x-087-2-firewall.html.

2.4.15.1 Basic Packet Filtering with iptables

Since iptables is a newer method, more has been written about ipchains, so iptables will be discussed here.

When developing server packet-filtering rules, the philosophy to follow is: if it's not used, don't open it up. No web server, don't open up port 80; no telnet, don't open port 23, and so on. Accordingly, start by closing off everything, then slowly opening services up on an as-needed basis.

a) _____ remove ipchains rules, kernel module and package

Out of the box, standard Red Hat Linux 7.2 installs and activates some ipchains rules. First, disable the run control script for subsequent system starts, then remove the rules and unload the ipchains kernel module.

```
# chkconfig --level 0123456 ipchains off
# /etc/rc.d/init.d/ipchains stop
# rmmod ipchains
```

Now the iptables kernel module can be added.

Additionally, consider removing the ipchains package. Since the 'lokkit' package depends upon the ipchains package, it will need to be removed first. The lokkit utility allows for creating simple firewall rules, and since more complex rules will be established manually, the lokkit utility is unneeded. Removing these packages reduces confusion about which firewall rule utility is used.

```
# rpm -e lokkit
# rpm -e ipchains
```

b) _____ get rid of any and all existing definitions

Start by flushing the iptables.

```
# iptables --flush
```

```
# iptables --flush -t nat
```

TEST: At this point all pings (localhost, remotehost) will fail with the error message "ping: sendto: Operation not permitted".

c) _____ deny all traffic

Next deny all traffic.

```
# iptables --policy INPUT   DROP
# iptables --policy OUTPUT  DROP
# iptables --policy FORWARD DROP
```

→ all pings will still fail.

d) _____ allow all traffic on internal interface

Allow all traffic on lo, the loopback interface.

```
# iptables -A OUTPUT -j ACCEPT -o lo
# iptables -A INPUT  -j ACCEPT -i lo
```

TEST: at this point a `ping localhost` should be successful.

e) _____ allow all outgoing conversations

```
# ST_NER="--state NEW,ESTABLISHED,RELATED"
# iptables -A OUTPUT -p tcp -m state $ST_NER -j ACCEPT
# iptables -A OUTPUT -p udp -m state $ST_NER -j ACCEPT
```

TEST: at this point an outgoing `telnet` to a remote node should be successful (provided that remote node grants access in `hosts.allow` and telnet service is enabled).

f) _____ allow basic ICMP conversations

```
# iptables -A INPUT -p icmp --icmp-type echo-reply \
      -s $MYNET -j ACCEPT
# iptables -A INPUT -p icmp --icmp-type destination-unreachable \
      -s $MYNET -j ACCEPT
# iptables -A INPUT -p icmp --icmp-type time-exceeded \
      -s $MYNET -j ACCEPT
```

This configuration allows some fundamental ICMP conversations, such as outgoing `ping`s, but does not allow incoming `ping`s (pong). Additionally, a `traceroute` to this system will also fail due to disallowing incoming `ping`s.

g) _____ allow incoming DNS resolution

```
# iptables -A INPUT  -p udp -m state $ST_NER --sport 53 -j ACCEPT
```

The above configuration will allow DNS resolution responses into the system. This can be further limited with the **`--source`** option restricting incoming messages to the official DNS servers.

TEST: DNS lookup should resolve successfully:

```
$ nslookup <valid-dns-entry>
```

h) _____ allow incoming SSH conversations

```
# iptables -A INPUT  -p tcp -m state $ST_NER --dport 22 -j ACCEPT
# iptables -A INPUT  -p udp -m state $ST_NER --dport 22 -j ACCEPT
```

The above configuration will allow SSH conversions from anywhere; to limit this to a specific host or network (all hosts starting with 192.168, in this example), include the option:

```
--source 192.168.1.1/24
```

for whichever appropriate source IP address/es. An address of 0/0 will grant access from any host.

i) _____ allow additional incoming service conversations

```
# iptables -A INPUT  -p tcp -m state $ST_NER --dport portno -j ACCEPT
# iptables -A INPUT  -p udp -m state $ST_NER --dport portno -j ACCEPT
```

The script "iptables_config", listed in the Appendix, can be used to set the above values in Steps b) through g). The script can be configured with a list of services to enable, effectively repeating Step i) for each listed service.

In this script, define the two variables:

`MYNET` network specification to allow incoming services from.

`ALLOWSERVICES` listing of services to enable.
list services to enable by "portnumber/protocol"
where: *portnumber* is the numeric port number, and
protocol is either "tcp", "udp", or "both".

This script is rather simple in that it grants the list of services to the same source networks. This can be customized individually per which source networks/hosts are allowed specific services.

j) _____ assert iptables configuration on system starts

One way to assert this customized iptables configuration is to include an RC script following the system iptables configuration `S08iptables`. The script in Appendix A is configured to insert itself as `S09iptables_config`. To install this RC script:

```
# chmod +x /etc/rc.d/init.d/iptables_config
# chkconfig iptables_config on
# /etc/rc.d/init.d/iptables_config start
```

2.4.15.2 Enabling Individual Services

For each service this server is going to provide, the ports necessary for that server will need to be opened up, like the incoming SSH connection example above in step e). Below is an initial list; refer to `/etc/services` and the output of "**`netstat -vatup`**" to track down other ports. Reference Section 5.4.2 for further detail on tracking down individual services with `netstat`, `ps` and `lsof`.

service	destination port/s	tcp?	udp?	recommended enabling?
ftp	20, 21	yes	no	only if ftp server
ssh	22	yes	no	yes
telnet	23	yes	no	no, unless telnet is required
mail	25	yes	no	no
http	80	yes	no	only if web server
ntp	123	no	yes	yes
samba	137, 138, 139	yes	yes	only if samba server
nfs	111, 1024, 1025, 1026, 1027, 2049	yes	yes	only if NFS server
receive syslog	514	no	yes	only if central logging server

service	source port/s	tcp?	udp?	recommended enabling?
receive DNS resolutions	53	no	yes	yes, if will be conducting DNS queries

2.4.15.3 More Features of ipchains

Additional features of ipchains:

logging: iptables can be configured to log events.

WARNING: These log files can grow *very rapidly*.

```
# iptables -A INPUT -j LOG -m limit --limit 20/minute \
    --log-prefix "iptables drop: "
```

stateful inspection: decisions can be based upon previous connection history (if any), by remembering such information as source and destination addresses, port numbers, header flags status, and TCP sequencing.

rate-limited matching: can limit the rate which the packet filter handles connection requests. This feature is useful under a "Denial of Service" (DoS) attack.

More in-depth discussions can be found at:
man page for iptables (8)
http://www.linuxsecurity.com/resource_files/firewalls/IPTables-Tutorial/iptables-tutorial/iptables-tutorial.html
http://www.linuxworld.com/site-stories/2001/0920.ipchains.html
http://netfilter.samba.org/unreliable-guides/packet-filtering-HOWTO/

2.4.16 Secure the Filesystem

Server security can be enhanced by mounting certain filesystem partitions read-only. By default, all local filesystem partitions are mounted read-write. However, through the course of day-to-day system operation, there are some file systems which do not need to be written to and can be mounted read-only to prevent the filesystem content from changing. Non-writeable partitions prevent the introduction of Trojan Horses into the filesystem and can safeguard from accidental deletions. Candidates for mounting read-only include `/usr` and `/usr/local`.

Other file systems (like `/` and `/var`) need to be read-write for normal system operation. Even if `/var` is relocated to a separate file system, root must still be read-write so `/etc` file contents can be updated during normal system operation (one example: `/etc/ntp/drift`); as `/etc` is expected to be part of the root file system, it couldn't be mounted separately to allow root to be mounted read-only.

a) _____ make `/usr` and `/usr/local` read-only

→ edit `/etc/fstab`, change:

```
device      /usr        ext3      defaults    x x
device      /usr/local  ext3      defaults    x x
```

to:

```
device      /usr        ext3      ro          x x
device      /usr/local  ext3      ro          x x
```

Know that making these filesystems read-only will require additional steps to update them later. The partition will need to be remounted read-write in order to modify the contents of these partitions:

```
# mount -o rw -o remount /usr
# mount -o rw -o remount /usr/local
```

And after updating these partitions, mount the read-only again:

```
# mount -o ro -o remount /usr
# mount -o ro -o remount /usr/local
```

The current read-write / read-only state of the filesystem/s can be found with the mount command:

```
# mount
```

and looking for the flags (ro) or (rw) near the end of output line.

TEST:

```
# mount -o rw -o remount /usr
# touch /usr/FOO                (touch is successful)
# mount -o ro -o remount /usr
# touch /usr/FOO
touch: creating '/usr/FOO': Read-only file system
# mount -o rw -o remount /usr
# touch /usr/FOO                (touch is successful)
# mount -o ro -o remount /usr
# touch /usr/FOO
touch: creating '/usr/FOO': Read-only file system
```

While the intent of making a filesystem read-only is to protect a system from having Trojan Horses installed on it, it is possible to remount read-write a currently mounted read-only filesystem. There may be some UNIXes where this is not possible, but it is possible with Red Hat Linux 7.2. Indeed many UNIX operating systems will initially mount root and `/usr` in a read-only fashion, then remount them read-write after passing an `fsck` check.

Due to this ability to remount read-write a currently read-only filesystem, this is only a slight extra measure for security: it protects more against inadvertent writing than to the filesystems than projecting against an installation of a Trojan Horse. In this manner, it is an extra layer of protection similar to Red Hat Linux 7.2 defining a root alias of '`rm -i`' to '`rm`'.

2.5 System Integrity Checking

System Integrity Checking is a process by which the system is audited to verify that everything is how it is supposed to be. Such an audit can highlight a variety of problems: wrong permissions and/or owners of key files and/or directories, locked faceless accounts now having passwords, to insertion of Trojan Horses, or other applications not appropriate for the system (such as an anonymous FTP server on a system that's not supposed to have FTP enabled).

A variety of both freely-available and commercial tools can provide most of the above functionality. Only you will know what applications are appropriate for a class of servers, or what accounts are and are not supposed to be on a system. For each class of system within your environment you will most likely need to prepare your own audit scripts to check for certain list of acceptable packages, and what accounts should always be locked.

2.5.1 File Integrity – Host Intrusion Detection

Tools such as AIDE (Advanced Intrusion Detection Environment) and Tripwire can run an audit of file integrity. With these utilities, a file system check is periodically conducted, comparing the current filesystem to a previously prepared snapshot of what the system is supposed to look like. This check will compare file characteristics: it can compare characteristics cursorily (file size, timestamps, checksum) to in-depth (file inode, more complex checksums).

These utilities will only work when there is snapshot of file characteristics of a known good system with which to compare. In order to guarantee the correctness of this snapshot database, this snapshot must be conducted before the system is deployed in the field (even before it is connected to the network) to ensure that nothing bad or incorrect has crept into it. Also, the database itself must be protected; if the both the contents of the filesystem and the file integrity database are identically updated, the change introduced into the filesystem will go unnoticed.

A file system integrity check can be employed as a host-based intrusion detection system. If, for instance, `/bin/login` starts changing on systems, there is either a patch update in progress or a Trojan Horse being is being deployed. Just because there is a change to the filesystem contents does not in and of itself mean there is an ongoing security incident – the reason for the change still needs to be investigated.

To reiterate, the initial system snapshot must be taken after everything has been installed and configured (all the package updates, too), then the snapshot taken *before* placing the system onto the network and putting it to use. For subsequent patch updates, the new snapshot database should be generated from a known good system – that is, one that is in the protected development and test environment. Before taking a new snapshot database, the test system should be validated to be clean (or better still: rebuild from original media), apply the update, and take a new snapshot.

2.5.1.1 Introduction to AIDE

AIDE is not part of the standard Red Hat Linux 7.2 release; it must be acquired and complied separately before configuring and running it.

a) _____ download AIDE and libgcrypt

AIDE: from http://www.cs.tut.fi/~rammer/aide.html (currently v0.8)
libgcrypt: from ftp://ftp.gnupg.org/gcrypt/alpha/libgcrypt/ (currently version 1.1.6)
Building this utility requires the bison, flex, and gcc packages.

b) _____ build libgcrypt

First build and install the prerequisite utility libgcrypt.

```
# gunzip libgcrypt-1.1.6.tar.gz
# tar xvfp libgcrypt-1.1.6.tar
# cd libgcrypt-1.1.6
# ./configure
# make
# make install
```

c) _____ build AIDE

Next build the AIDE utility.

```
# gunzip aide-0.8.tar.gz
# tar xvfp aide-0.8.tar
# cd aide-0.8
# ./configure
# make
# make install
```

This will install the aide binary in `/usr/local/bin`, and place the `aide.conf` file in `/usr/local/etc`. Add options to `configure` to change these locations ('`configure --help`' to list the options).

d) _____ create aide.conf

Create an `aide.conf` configuration file `/usr/local/etc/aide.conf`) to define what directories/files to check, and what characteristics of these directories/files to record.

The example `aide.conf` listed here is a modification of the example provided at the AIDE source web site. This configuration was turned to make it applicable to the directories and files in Red Hat Linux.

```
# example /usr/local/etc/aide.conf

   #p:      permissions
   #i:      inode
   #n:      number of links
   #u:      user
   #g:      group
   #s:      size
   #b:      block count
   #m:      mtime
   #a:      atime
   #c:      ctime
   #S:      check for growing size
   #md5:    md5 checksum
   #sha1:   sha1 checksum
   #rmd160: rmd160 checksum
   #tiger:  tiger checksum
   #R:      p+i+n+u+g+s+m+c+md5
   #L:      p+i+n+u+g
   #E:      Empty group
   #>:      Growing logfile p+u+g+i+n+S

 # You can also create custom rules - my home made rule
 # definition goes like this
```

```
MyRule = p+i+n+u+g+s+b+m+c+md5+sha1

# Next decide what directories/files you want in the database

/etc p+i+u+g     # check only permissions, inode, user and group for etc
/bin MyRule      # apply the custom rule to the files in bin
/sbin MyRule     # apply the same custom rule to the files in sbin
/usr/bin MyRule
/usr/include MyRule
/usr/kerberos MyRule
/usr/lib MyRule
/usr/libexec MyRule
/usr/local MyRule
/usr/sbin MyRule
/usr/share MyRule
/lib MyRule
/dev MyRule
/boot MyRule
/var MyRule
!/var/log/.*     # ignore some directories that change too often
!/var/spool/.*
!/var/tmp/.*
!/var/log/utmp$  # ignore the file /var/log/utmp
!/var/log/wtmp$  # ignore the file /var/log/wtmp
```

e) _____ initialize AIDE database

Create the initial AIDE database. This must be done while the system is known to be good, before it goes live on the network and users can log into it.

```
# aide --init
```

Running System Integrity Check

To inspect the file system integrity, run:

```
# aide --check
```

Updating the AIDE Database

If the AIDE database needs to be updated, run:

```
# aide --update
```

The AIDE database will need to be updated following a legitimate update to the system. First, run a check of the system to verify the integrity, conduct the package upgrade or configuration change, and then update the AIDE database.

2.5.1.2 Introduction to Tripwire

Tripwire v2.3.1.2 is included in the standard Red Hat Linux 7.2 distribution. This is the Open Source version of Tripwire; a commercial version is also available. The Open Source version from the Red Hat Linux 7.2 distribution is discussed here.

a) _____ verify tripwire is installed

```
# rpm -qa tripwire
```

b) _____ run tripwire installation script

Run the tripwire installation script to declare the site and local system keyfile passphrases.

```
# cd /etc/tripwire
# ./twinstall.sh
```

Select a strong passphrase. These passphrases must be more than 8 characters in length.

```
Enter the site keyfile passphrase:   enter-site-passphrase
Verify the site keyfile passphrase:  enter-site-passphrase

Enter the local keyfile passphrase:  enter-local-passphrase
```

```
Verify the local keyfile passphrase: enter-local-passphrase

Signing configuration file...
Please enter your site passphrase:   enter-site-passphrase
Wrote configuration file: /etc/tripwire/tw.cfg
```

A clear-text version of this file is stored at `/etc/tripwire/twcfg.txt`.

```
Signing policy file...
Please enter your site passphrase:   enter-site-passphrase
Wrote policy file: /etc/tripwire/tw.pol
```

A clear-text version of this file is stored at `/etc/tripwire/twpol.txt`.

c) _____ define tripwire policy file

The default tripwire configuration file works well only when all packages have been installed; using this default configuration will result in many errors due to this installation being trimmed to fit the purpose of the server. As indicated with the twinstall.sh output, this is only a minimal policy, and should be modified to describe the system and should be monitored.

```
# vi /etc/tripwire/twpol.txt
```

The policy file delivered with standard Red Hat Linux 7.2 lists many entries specifically by filename and groups these entries according to function and purpose.

Encrypt the updated policy file with

```
# twadmin --create-polfile twpol.txt
```

and enter the site passphrase as prompted.

d) _____ initialize tripwire database

```
# tripwire --init
```

and enter the local passphrase as prompted.

e) _____ protect policy and configuration files on production systems

The production systems should not have a clear-text copy of the Tripwire policy and configuration files. Retain a copy of the clear-text files *only* in the development environment; delete the clear-text copies on all other systems.

```
# rm /etc/tripwire/*.txt
```

If the clear-text copies are available on production systems, an intruder may discover a short-coming in the policy and take advantage of it. While the encryption of the policy and database files will protect against alteration, it will not protect against that which is not expressly checked for per the policy. Hiding the policy makes finding such a gap in the policy pure guess-work.

Running System Integrity Check

To inspect the file system integrity, run:

```
# tripwire --check
```

Updating the Tripwire Database

If the Tripwire database needs to be updated, run 'tripwire --update'.

```
# tripwire --update
```

The Tripwire database will need to be updated following a legitimate update to the system. First, run a check of the system to verify the integrity, conduct the package upgrade or configuration change, and then update the Tripwire database.

Updating the Tripwire Policy File

If the Tripwire policy file needs to be updated, run:

```
# twadmin --create-polfile twpol.txt
# tripwire --init
```

to update the encrypted policy file and recreate the Tripwire database according to the new policy.

2.5.1.3 Protecting the Audit Snapshot Database

The audit database itself needs to be protected. An audit tool is only as good as what it is comparing against. If that audit database is compromised, the audit tool can then give erroneous results, allowing a break-in to go undetected. Traditionally the audit database itself could be stored on read-only media (making a CD-ROM, or setting the read-only jumper on a hard drive directly), or verified (or refreshed) over-the-wire from a known good and protected source. With the release of Tripwire provided with Red Hat Linux 7.2, this database is also protected with encryption. If the database file itself is modified the database signature will not match and the database compromise detected.

2.5.2 Virus Scanning

There are commercial anti-virus software products that can be purchased and run on Linux systems. Mostly these will check for Windows-type virus fingerprints on the system. This can be important to do if the system is sharing filesystems with Windows systems through Samba. Conducting a virus scan locally is much more I/O-effective than doing so over through an exported file system remotely mounted on a Windows system.

2.5.3 Comparing Virus Scanning to File Integrity Checking

Virus scanning differs from file integrity checking. The virus scanner compares to a database of known virus fingerprints, with the fingerprint database prepared by an external party. A file integrity checker compares the current filesystem characteristics to the characteristics of a known previously good file system, and notes the differences in the characteristics of the files. With a virus scanner, the virus itself will likely be identified; whereas with a file integrity checker only the difference to the filesystem is noted, but the reason for the difference still needs to be determined.

These two utilities do not strictly compete with each other, rather complementing each other. Perhaps a virus scanner may know about a particular Trojan Horse for `/bin/ls`. The virus scanner will notice the fingerprint of this Trojan Horse and will flag it as a system infection (and likely providing the virus name, giving leads to system clean-up procedures). The file integrity checker will note that `/bin/ls` has been changed, but does not provide further detail as to what the nature of the corrective action will be.

---

3 Hardening Server-Specific Services

This section lists steps to harden the configuration of services that would be specific to the purpose of a machine – these services should not be applied universally to all servers across the infrastructure.

The steps detailed here should be applied to a server following application of the general security steps detailed above in Section 2.4. These server-specific configuration steps are to be applied in addition to (and sometimes removal of) the general security steps.

3.1 Network Configuration

The network layout between the FTP and syslog servers can contribute to security. With a "flat" network, in Figure 1, all systems are connected to the same network and face the public Internet. Each system is equally vulnerable to probes, scans and attacks from the open Internet.

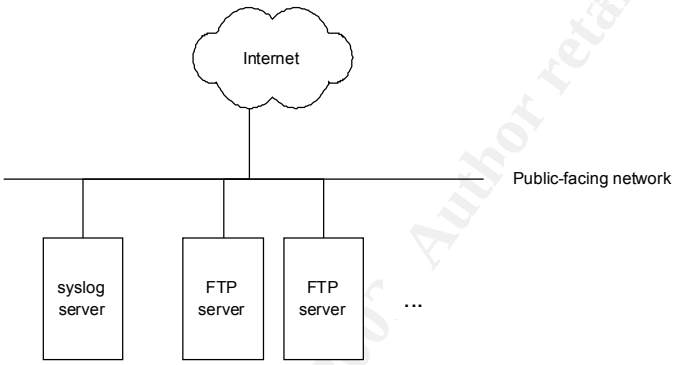

Figure 1: Flat Network Diagram

Due to the nature of an anonymous FTP server being a public access system, it cannot be protected on a private network. However, the syslog server can. Figure 2 introduces a simple network layout modification: adding a private network behind public-facing servers to interface these servers with "behind the scenes" servers such as the syslog server. With a private network, connectability to these systems and its network is limited to internal only, immediately improving security. For the host to be compromised from outside and connected to (or its network snooped), it must be done so through a public-facing server first.

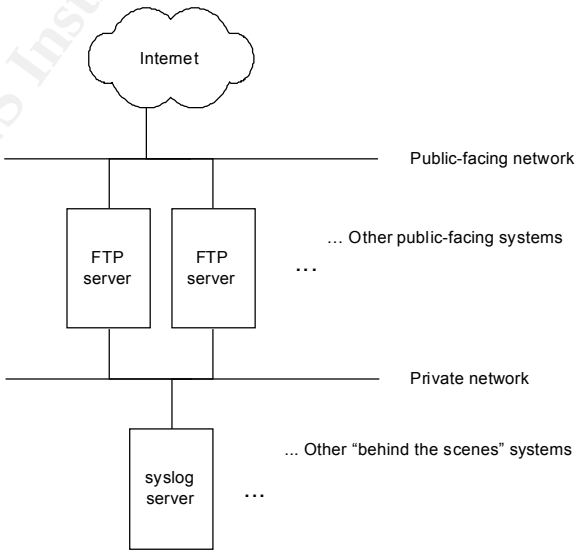

Figure 2: Private Network for Non Public-Facing Systems

3.2 Configuring FTP Server

FTP is a service which may not be necessary for the purpose of the system. If the purpose of the server an anonymous FTP server, this obviously is not the case. For user-based FTP service from entirely within the site, the `ftp` executable can be removed and replaced with `sftp`, a secure FTP client of the SSH suite.

Standard Red Hat Linux 7.2 comes with the wu-ftpd package. This package frequently has updates to address security vulnerabilities. Before deploying the system, ensure the latest wu-ftpd version is installed. Refer to the updates.redhat.com site mentioned above in Section 2.3.3, or find the latest version at the wu-ftpd site at http://www.wu-ftpd.org/.

a) _____ get the latest version of wu-ftpd
wu-ftpd frequently has security vulnerabilities patched with the latest version.
Red Hat Linux 7.2 media delivers wu-ftpd 2.6.1-18, but a newer version is already available.
Refer to Section 2.3.3.1 for how to get the latest version.

b) _____ restrict incoming services
Restrict from where FTP service can connect to with configuration in TCP wrappers and/or in `xinetd.conf`, as described previously.

3.2.1 in.ftpd Arguments

Consider some of the options for the FTP daemon:

`-a` / `-A`	`-a` activate `ftpaccess` for access control. Highly recommended. `-A` completely ignore `ftpaccess` for access control (default).
`-l`	log each ftp connection. Highly recommended.
`-L`	log each ftp command. Note that if a user types their password at the wrong time, their password will be logged. Instead of an option to `wu-ftpd`, to establish the same functionality in `/etc/ftpaccess`, add the configuration entry: `log commands anonymous,guest,real`
`-t`	specify timeout, in seconds. Default is 15 minutes (900 seconds). Consider decreasing this value to drop the unattended network connections more quickly. This may be be necessary for high-volume FTP servers. Too high and the number of concurrent connections may be too high for the system; too low and the FTP service may be considered rude by its customers.
`-I`	log each put request. Recommended, given past history of ftpd bugs; logging put requests to a remote secure server allows reconstruction of what happened and how in the event of an incident.
`-o`	similar to `-i`, but for logging output.
`-X`	modifies `-i` and `-o` to log via syslog. Recommended for with a large quantity of FTP servers, so logging is centralized.
`-w` / `-W`	`-w` Record every login and logout in `wtmp` file (default). `-W` Specified user logins are not recorded in the `wtmp` file.
`-q` / `-Q`	`-q` Use PID files (default). `-Q` Don't use PID files. PID files are required by limit directive to determine the number of current users in each access class. Disabling PID files will disable user limit control.

For the FTP daemon initiated through xinetd (default configuration with Red Hat Linux 7.2), daemon options are defined in `/etc/xinetd.d/wu-ftpd`. Default options are "`-l`" and "`-a`".

3.2.2 Disabling Anonymous FTP

For a system that will not provide anonymous FTP, this service should be removed.

a) _____ remove anonymous FTP package

```
# rpm –e anonftp
```

This packages delivers the files and account necessary to enable to anonymous FTP.

b) _____ remove anonymous, guest setting from ftpaccess

→ edit `/etc/ftpaccess`, remove entries "anonymous" and "guest" from the class definition

```
class       all          real         *
```

The other entries of "anonymous" and "guest" – commands chmod, delete, …; and log specifications – should remain, in the event that anonymous ftp is enabled at a later time.

3.2.3 Configuring Anonymous FTP

If anonymous FTP is necessary, provide this anonymous FTP service on a server separate from other servers. For stronger security, place the anonymous FTP service on a system that is strictly an anonymous FTP server; do not pair up an anonymous FTP service with other services on the same server.

a) _____ verify ftp account

Verify the `ftp` account exists. With out this account, the system will not allow anonymous FTP.

```
# grep ftp /etc/passwd
```

```
ftp:x:50:FTP User:/var/ftp:/sbin/nologin
```

b) _____ specify ftp's home directory

The FTP users entry in `/etc/passwd` specifies the home directory. The standard Red Hat Linux 7.2 package "anonftp" delivers "`/var/ftp`" as ftp's home directory. If this home directory is relocated, also relocate the contents of `/var/ftp` to the new home directory; there are necessary files in the bin and lib directories to allow anonymous FTP to function. To move `/var/ftp` to `/home/ftp`:

```
# cd /var
# tar cvf ftp.tar ftp
# cd /home
# tar xvfp /var/ftp.tar
# rm /var/ftp.tar
```

edit `/etc/password`, changing the home directory for the ftp account from:

```
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
```

to:

```
ftp:x:14:50:FTP User:/home/ftp:/sbin/nologin
```

Note that if FTP's home directory is changed, and the anonftp package later updated, the home directory of the FTP account will be changed back. Also, any content updates will be written to `/var/ftp`, and will need to be copied into the proper FTP home directory.

c) _____ verify ftp user is nologin and password is locked

→ verify "`/sbin/nologin`" for user shell

```
# grep ftp /etc/passwd
```

```
ftp:x:14:50:FTP User:/home/ftp:/sbin/nologin
```

→ verify password is locked (second field is a * or !!)

```
# grep ftp /etc/shadow
```

```
ftp:*:…
```

d) _____ verify contents of FTP's passwd, and group files

Anonymous ftp has it's own password and group files, which should *not* be a copy of the system's `/etc/passwd` and `/etc/group`. These collections of files should be separate and distinct.

FTP's passwd and group files are stored in `~ftp/etc`, and are only referenced for an owner and group lookup when the anonymous ftp user lists the directory contents. If the directory listing contains the actual user and/or group, the user and/or group can be listed in the `passwd` and `group` files, do leave the other fields empty (shell, full name, directory). That way, if FTP's password file are obtained, the information contained within cannot be used to compromise the system accounts. However, providing a list of user names will provide an attacker with a list of valid user names with which to attempt guessing passwords with.

The standard Red Hat Linux 7.2-provided `~ftp/etc/passwd` and `~ftp/etc/group` contain the following entries, and are generally acceptable:

→ verify `~ftp/etc/passwd` entries

```
root:*:0:0:::
bin:*:1:1:::
operator:*:11:0:::
ftp:*:14:50:::
nobody:*:99:99:::
```

→ verify `~ftp/etc/group` entries

```
root::0:
bin::1:
daemon::2:
sys::3:
adm::4:
ftp::50:
```

e) _____ set home directory permissions

```
# cd ~ftp
# chown root:ftp pub
# chmod 2555 pub                (set-gid, read and execute only, no write)
# chown –R root:root bin etc lib
# chmod 0111 bin etc            (execute only)
# chmod 0555 lib                (read and execute only, no write)
# chmod 0555 bin/* lib/*        (read and execute only, not write)
# chmod 0444 etc/*              (read only)
```

f) _____ log file transfers

Verify that file transfers are logged.

→ check `/etc/ftpaccess`, look for entry, or add it if it doesn't exist.

```
log transfers anonymous,guest,real inbound,outbound
```

g) _____ define anonymous access class

Establish `ftpaccess` class which allows anonymous ftp access.

→ add entry to `/etc/ftpaccess`

```
class   anonftp  anonymous  *
```

h) _____ create `~ftp/welcome.msg`

The `welcome.msg` is a text file displayed after an anonymous ftp user connects to the service. This is similar to `/etc/motd` after a user has logged in. Create an appropriate "acceptable use / no unauthorized use" text message here.

```
# chmod 0444 ~ftp/welcome.msg
```

i) _____ protect `lost+found` directories on separate filesystems

If some of the FTP directories are maintained as separate file systems, extra consideration must be given to the lost+found directory of that file system to prevent inadvertent file retrieval capability in the event of a corrupted file system.

```
# disallow file retrieval of anything in lost+found, in case
# it is maintained as a separate filesystem.
noretrieve  /home/ftp/lost+found
noretrieve  /home/ftp/incoming/lost+found
noretrieve  /home/ftp/pub/lost+found
```

j) _____ protect incoming directory

If an incoming directory is needed, extra attention is required to its configuration. In general it is never a good idea to allow write access to an anonymous directory; it can become a denial of service vector by filling up a disk, or could become a "warez" site for distributing pirated software.

The incoming FTP directory should be its own separate partition. In this way, if the incoming directory is filled to capacity, it will not hamper other normal system operation. If the incoming FTP directory were part of a critical directory, it could be used as a denial-of-service attack against the system.

1. _____ create FTP administrator account, to manage files in incoming directory:

   # **vi /etc/passwd**

   ```
   ftpadmin:x:16:16::/home/ftp:/bin/bash
   ```

   # **vi /etc/shadow**

   ```
   ftpadmin:*::::::::
   ```

   # **vi /etc/group**

   ```
   ftpadmin:x:16:ftpadmin
   ```

2. _____ create the incoming directory, with write, but not read, access:

   # **mkdir –m 3773 ~ftp/incoming**

   If users are running GUI ftp clients (such as Netscape), the permissions may need to be 777 on the incoming directory instead.

   # **chown ftpadmin:ftpadmin ~/ftp/incoming**

   Set the directory owner and group to the FTP administrator account.

3. _____ protect FTP home directory and incoming directory from on-the-fly tar'ing:

   # **cd ~ftp/.notar**
   # **touch .notar incoming/.notar**
   # **chmod 0000 .notar incoming/.notar**

   Add to `/etc/ftpaccess`:

   ```
   noretrieve  .notar
   ```

   This "`noretrieve .notar`" entry declares the `/home/ftp/.notar` file as non-retrievable, as some web clients and FTP proxies can be confused by the `.notar` file.

4. _____ edit `/etc/ftpaccess` to include the following entries, preventing guest and anonymous uses from conducting the following actions:

   ```
   chmod            no    guest,anonymous      # chmod permission?
   umask            no    guest,anonymous      # umask permission?
   delete           no    guest,anonymous      # delete permission?
   overwrite        no    guest,anonymous      # overwrite permission?
   rename           no    guest,anonymous      # rename permission?
   ```

5. _____ restrict activity to in the incoming directory, adding these entries to /etc/ftpaccess:

```
path-filter anonymous  /etc/pathmsg ^[-A-Za-z0-9._]*$ ^\. ^-
upload      /home/ftp  *  no
noretrieve  /home/ftp/incoming
```

The "path-filter" line restricts filenames of uploaded files to letters, numbers, hyphen, underscore, and a period. It also restricts a file from starting with a period or hyphen. If the filename conditions are not followed, the message in ~ftp/etc/pathmsg will be displayed. This should be a text file explaining what the upload filename conditions for the server are.

The "upload" line disallows uploads into the FTP home directory. This line is a default, but include it so it isn't forgotten about while debugging.

The "noretrieve /home/ftp/incoming" line means that once a file is written there, it cannot be retrieved. The incoming directory should be monitored closely. With this "noretrieve" option, the only method for subsequent use of the file is for the FTP administrator to relocate it first. Before doing so, the file content should be inspected, such that it is as it is advertised as.

To Disallow Sub-Directory Creation in /incoming

```
# disallow subdirectory creation in incoming directory.
# allow file uploads to just incoming directory.
upload  /home/ftp  /incoming    yes  ftpadmin ftpadmin 0440 nodirs
```

This "upload" clause specifies that files uploaded into the incoming directory will be assigned user and group id of the FTP administrator and have permission mode 0440.

To Allow Sub-Directory Creation in /incoming

If anonymous FTP users are to be allowed to create subdirectories under the incoming directory, replace the second "upload" clause above with the following two entries:

```
# allow 1 subdirectory creation, immediately in incoming.
# allow file uploads to incoming dir, or immediate subdirectory.
upload  /home/ftp  /incoming    yes  ftpadmin ftpadmin 0440 dirs 3773
upload  /home/ftp  /incoming/*  yes  ftpadmin ftpadmin 0440 nodirs
```

With this configuration, only one subdirectory level can be created. When created, the directory will be assigned permission mode 3773. Files created in the incoming directory or in a subdirectory, will have the uid/gid of ftpadmin and permission 0400.

k) _____ run FTP daemon stand-alone

With the configuration provided thus far, the FTP daemon is initiated through xinetd. For an anonymous FTP server, consider running FTP as a standalone daemon. Moving ftpd out of xinetd allows for the removal of xinetd altogether, further minimalizing the server. Additionally, there will be a performance gain with a standalone ftpd: each incoming FTP request is handled immediately by ftpd, instead of xinetd fork() ing itself and exec() ing in.ftpd.

1. _____ edit /etc/xinetd.d/wu-ftpd to disable ftpd in xinetd:
   Change the wu-ftpd entry of disable to:

```
disable = yes
```

   And restart xinetd to make the change immediate:
   # **/etc/rc.d/init.d/xinetd restart**

2. _____ create FTP daemon run-control script:

→ create /etc/rc.d/init.d/ftpd
Full script detail is located in Appendix A.

→ install the run control script:

```
# chown root:root /etc/rc.d/init.d/ftpd
# chmod 0555 /etc/rc.d/init.d/ftpd
# chkconfig --add ftpd
# chkconfig ftpd on
# /etc/rc.d/init.d/ftpd start
```

3.2.4 Service Minimalization for Anonymous FTP Server

For systems that will be an anonymous FTP server, FTP should be the only service provided. Some of the services configured in the general-purpose security steps above in Section 2.4 should be removed.

a) _____ iptables configuration

From §2.4.15: grant incoming services only for DNS, SSH, NTP and FTP.

No other incoming services should be granted through `iptables`.

b) _____ TCP Wrappers configuration

From §2.4.11: Configure FTP and SSH services.

Open FTP service to all.

`/etc/hosts.allow`:

```
in.ftpd: ALL
```

Grant incoming connections (SSH) to a select small list of permitted hosts, such as only your workstation.

`/etc/hosts.allow`:

```
sshd:    LOCAL, localhost, mysystem.mydomain.com
```

c) _____ xinetd configuration

From §2.4.12: disable all `xinetd` based services except for the FTP daemon.

If the FTP daemon is configured to run as stand-alone daemon, as provided in Section 3.2.3, step k), FTP configuration in `xinetd` must be disabled. With all other `xinetd` based services disabled, it is possible to remove the `xinetd` package altogether. If the package is removed, it will be impossible to enable any `xinetd` based service without first installing the `xinetd` package.

```
# rpm –e xinetd
```

3.2.5 Testing the Anonymous FTP Server

The implementation of the Anonymous FTP server should be tested to verify that it is configured and functioning as it is intended to be. Testing should include:

1. ability to connect as anonymous user.
2. display of message files (`welcome.msg`, `etc/pathmsg`) as appropriate.
3. ability to browse and retrieve from `/pub` directory.
4. ability to upload, but not retrieve, from `/incoming` directory [if configured as such].

With the test sequence displayed here, `/etc/ftpaccess` was configured for:

a. uploading to `/incoming` only.
b. allow creating of one subdirectory under `/incoming`.
c. no file retrieval or directory viewing of `/incoming` contents.

TEST:

1. _____ verify that `~ftp/welcome.msg` content is displayed after logging in:

```
$ ftp localhost
Connected to localhost (127.0.0.1).
```

```
220 FTP server ready.
Name (localhost:username): ftp
331 Guest login ok, send your complete e-mail address as password.
Password: me@localhost      (password will not be echoed back)
230-The response 'me@localhost' is not valid
230-Next time please use your e-mail address as your password
230-        for example: joe@intruder
230-Authorized use only.  Unauthorized use is not permitted.
230-All logins and file transfers will be logged.
230-If you do not like this policy, disconnect now.
230-
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

2. _____ verify that `.notar` file is not displayed:

```
ftp> dir
227 Entering Passive Mode (127,0,0,1,39,165)
150 Opening ASCII mode data connection for directory listing.
total 48
d--x--x--x   2 root     root         4096 <time-stamp> bin
d--x--x--x   2 root     root         4096 <time-stamp> etc
drwxrwx-wx   6 16       16           4096 <time-stamp> incoming
dr-xr-xr-x   2 365      root         4096 <time-stamp> lib
dr-xr-xr-x   3 root     50           4096 <time-stamp> pub
-r--r--r--   1 root     root          153 <time-stamp> welcome.msg
226 Transfer complete.
ftp>
```

3. _____ verify that no directory content is displayed:

```
ftp> dir incoming
227 Entering Passive Mode (127,0,0,1,141,3)
150 Opening ASCII mode data connection for directory listing.
total 0
226 Transfer complete.
ftp>
```

4. _____ verify that "`.notar is marked unretrievable`" is displayed:

```
ftp> get .notar
local: .notar remote: .notar
227 Entering Passive Mode (127,0,0,1,63,146)
550 /.notar is marked unretrievable
ftp>
```

5. _____ verify no write permission to FTP's root directory:

```
ftp> put /etc/motd motd
local: /etc/motd remote: motd
227 Entering Passive Mode (127,0,0,1,36,25)
553 motd: Permission denied on server. (Upload)
ftp>
```

6. _____ verify ability to write to incoming directory:

```
ftp> cd incoming
250 CWD command successful.
ftp> put /etc/issue issue
local: /etc/issue remote: issue
227 Entering Passive Mode (127,0,0,1,19,132)
150 Opening BINARY mode data connection for issue.
226 Transfer complete.
35 bytes sent in 0.000161 secs (2.1e+02 Kbytes/sec)
ftp>
```

7. _____ verify file existence, owner, group, and permissions:

```
# cd ~ftp/incoming
# ls -la issue
-r--r-----    1 ftpadmin ftpadmin       35 <time-stamp> issue
```

8. _____ verify that `incoming/issue` is not retrievable:

```
ftp> get issue
local: issue remote: issue
227 Entering Passive Mode (127,0,0,1,225,133)
550 /incoming/issue is marked unretrievable
ftp>
```

9. _____ verify that `incoming/subdir` directory is created, check `subdir` permissions:
   Note that owner and group of the created subdirectory will be root, not ftpadmin.

```
ftp> mkdir subdir
257 "/incoming/subdir" new directory created.
ftp>
```

```
# cd ~ftp/incoming
# ls -lad subdir
drwxrws-wt    2 root     root         4096 <time-stamp> subdir
```

10. _____ verify file uploaded to correct location, with correct file owner, group and permissions:

```
ftp> cd subdir
250 CWD command successful.
ftp> put /etc/issue issue
local: /etc/issue remote: issue
227 Entering Passive Mode (127,0,0,1,36,242)
150 Opening BINARY mode data connection for issue.
226 Transfer complete.
35 bytes sent in 0.000114 secs (3e+02 Kbytes/sec)
ftp>
```

```
# cd ~ftp/incoming/subdir
# ls -la
total 12
drwxrws-wt    2 root     root         4096 <time-stamp> .
drwxrws-wt    7 ftpadmin ftpadmin     4096 <time-stamp> ..
-r--r-----    1 ftpadmin ftpadmin       35 <time-stamp> issue
```

11. _____ verify second level subdirectory creation not allowed:

```
ftp> cd /incoming/subdir
250 CWD command successful.
ftp> mkdir subdir
550 subdir: Permission denied on server. (Upload dirs)
ftp>
```

12. _____ verify that file upload with improper filename not allowed:
13. _____ verify message from `~ftp/etc/pathmsg` is displayed with improper filename:

```
ftp> cd /incoming
250 CWD command successful.
ftp> put /etc/issue .issue
local: /etc/issue remote: .issue
227 Entering Passive Mode (127,0,0,1,11,105)
550-~ftp/etc/pathmsg
550 .issue: Permission denied on server. (Filename (deny))
ftp>
```

14. _____ verify that file upload outside of `~ftp/incoming` is disallowed:

```
ftp> cd /pub
```

```
250 CWD command successful.
ftp> put /etc/issue newupload
local: /etc/issue remote: newupload
227 Entering Passive Mode (127,0,0,1,255,182)
553 newupload: Permission denied on server. (Upload)
ftp>
```

15. _____ verify that contents of ~ftp/pub are viewable:

```
# cd ~ftp/pub
# rm -rf
# cp /etc/issue issue
# ls -la
total 12
dr-xr-sr-x    2 root     ftp          4096 <time-stamp> .
drwxr-xr-x    7 root     root         4096 <time-stamp> ..
-r--r--r--    1 root     ftp            35 <time-stamp> issue

ftp> cd /pub
250 CWD command successful.
ftp> dir
227 Entering Passive Mode (127,0,0,1,250,18)
150 Opening ASCII mode data connection for directory listing.
total 8
-r--r--r--   1 root     50             35 <time-stamp> issue
226 Transfer complete.
ftp>
```

16. _____ verify that contents of ~ftp/pub are downloadable:

```
ftp> get issue /tmp/issue
local: /tmp/issue remote: issue
227 Entering Passive Mode (127,0,0,1,225,69)
150 Opening BINARY mode data connection for issue (35 bytes).
226 Transfer complete.
35 bytes received in 0.000238 secs (1.4e+02 Kbytes/sec)
ftp>
```

3.3 Configuring syslog Server

A central syslog server (or "loghost") is an important and useful tool for maintaining secure event and audit logs. Individual hosts throughout the site will record syslog messages locally on that host, and also forward messages to the central log server. This central syslog server gives two benefits: a central location from which to inspect the logs, and a duplicate copy (presumably more secure) of the logs. While an individual host may become compromised and its log files altered or pruned to cover the trail of the intrusion, the event history can still be recalled if a copy of those logs are stored elsewhere.

A central log server is also a key access point for monitoring events that report into syslog – disk full monitors, run away process monitors, non-existent process monitors. Utilities such as `logcheck` and `swatch` can be used to watch for security violations and other unusual activity. Tools could be set up to run reports on a scheduled interval, or established to provide real-time alerts to a monitor board.

3.3.1 Disk Configuration

System needs lots of online disk space to retain the log entries. Consider mounting `/var/log` as its own drive, or changing the logging directory to something other than `/var/log`, perhaps to `/home/syslog`, but also on its own filesystem.

a) _____ configure with lots of disks space

The configuration introduced and discussed in Sections 2.2.2.2 and 2.2.2.3, mounts `/var/log` as a separate partition on a separate disk.

3.3.2 syslog Server Backups

Syslog servers need regular, reliable backups to keep log histories. Tapes should not be recycled, but stored indefinitely in the event that the logs from a past event need to be referenced. Even more ideal, these logs could be archived on a write-once media, such as CD-R, for complete integrity and unalterability.

3.3.3 Service Minimalization for syslog Server

Since this system is to be used for event history and, if necessary, event recollection, this system must be very secure. To assist in attaining this higher security, it should *only* be a central log server and provide no other function.

a) _____ iptables configuration

From §2.4.15: grant incoming services only for NTP, incoming syslog, and SSH.
No other incoming services should be granted through `iptables`.

b) _____ install syslog

The syslog server should run only syslog, NTP client, and, possibly, SSH. It should be stripped down of other services and their packages deleted.

c) _____ install NTP client

NTP client services are necessary to keep the system time synchronized with the systems reporting to it.

d) _____ install SSH

If remote access into the system is to be permitted, it should be done so through SSH. SSH should be configured to grant the service only from a select few trusted hosts. Even SSH should be considered for removal, requiring access from just the console or a hardwired serial connection.

Just as with the FTP server, grant incoming connections (SSH) to a select small list of permitted hosts, such as only your workstation.

`/etc/hosts.allow`:

```
sshd:    LOCAL, localhost, mysystem.mydomain.com
```

3.3.4 syslog Configuration

a) _____ configure to accept remote log messages

To accept remote incoming log messages, syslog must be started with the "-r" switch. This should be defined in `/etc/sysconfig/syslog`, adding a "`-r`" into the `SYSLOGD_OPTIONS` setting.

```
SYSLOGD_OPTIONS="-m 0 -r"
```

Consider including the "`-x`" switch as well, disabling DNS name lookups. Doing so can protect against possible deadlock conditions between the DNS server and the loghost, and can also protect against faulty DNS resolutions, such as when the DNS server has also been compromised.

b) _____ configure to limit remote log messages from within the site

There is no built-in mechanism in syslogd or TCP wrappers to limit incoming log messages to a specified list of permitted hosts. This can be accomplished, however, with packet filtering (`ipchains` or `iptables`). Messages to syslog come through UDP port 514. Incoming syslog messages must be restricted to a set list of valid hosts to avoid a denial of service attack where the central log host is flooded with messages. Continuing with the `iptables` structures from earlier, this can be implemented by adding the below sequence to the run-control script `iptables_config`, first described in Section 2.4.15.1, Step j):

```
# iptables -A INPUT  -p udp --source 192.168.1.1/24 -m state \
    --state NEW,ESTABLISHED,RELATED --dport 514 -j ACCEPT
```

c) _____ configure log rotation

The `logrotate` program, configured in section 2.4.6 above, is designed to rotate, preserve, and eventually delete log files after a certain period of time or when the files reach a certain size. For a central syslog server, log rotation should be configured to preserve log files for a much longer interval.

To rotate files weekly and keep one years worth, edit `/etc/logrotate.conf` to:

```
# rotate log files weekly
weekly

# keep one years worth of backlogs
rotate 52
```

For a very busy syslog server, archival on a weekly basis may be necessary to keep the central syslog server file logs from growing too large.

To rotate files monthly and keep one years worth, edit `/etc/logrotate.conf` to:

```
# rotate log files monthly
monthly

# keep one years worth of backlogs
rotate 12
```

Archiving Logs

There are several strategies one could employ if log files are to be written to write-once media for permanent archival. The decision rests, in part, upon the capabilities of the write-once media.

The timeliness of the commit to the write-once media should be considered. During the time lag between the logs existing only on the hard disk of the syslog server and before it is committed to permanent archival, the logs can be considered at risk – vulnerable to alteration and deletion.

If the media can be written to several times in an append fashion, one could write syslog messages to the write-once media as they arrive (by configuring an entry in `syslog.conf` to write to the log file on the media), or as the logs are rotated (weekly, monthly, …). If written as messages arrive, care must be taken that messages are not while the media volumes are being exchanged.

The capacity of the media should also must be considered. If the syslog server is very busy and receives a lot of messages, the expected size of the log files at the time of rotation should be considered. If employing a writeable CD-ROM, capacity is roughly 650MB. If a months-worth of log files exceeds this size, consider rotating the logs and committing the logs to the media on a weekly basis. One wouldn't want to have the log files overrun the capacity of the media.

3.3.5 Testing the syslog Configuration

With the `syslog.conf` configuration outlined earlier in Section 2.4.6, syslog is configured for forward messages central syslog server with name of `loghost`. In this configuration, "*.warn" messages are written to the local `/var/log/syslog` file, and forwarded to the syslog server `loghost`. Also, "mail.*" are written the local `/var/log/maillog` file. Testing can be conducted utilizing the `logger` command.

TEST:

1. _____ verify syslog configuration:

   node# **grep warn /etc/syslog.conf**

   and verify the following entries:

```
*.warn;*.err          /var/log/syslog
*.warn;*.err          @loghost
```

   node# **grep mail /etc/syslog.conf**

and verify the following entry:

```
mail.*                  /var/log/maillog
```

2. _____ verify syslog message sent to local files and to syslog server `loghost`:

```
node$ logger -p user.warn "test message user.warn"

node# cd /var/log ; tail -1 messages ; tail -1 syslog
timestamp node username: test message user.warn
timestamp node username: test message user.warn

loghost# cd /var/log ; tail -1 messages ; tail -1 syslog
timestamp loghost username: test message user.warn
timestamp loghost username: test message user.warn
```

3. _____ verify mail log message sent to local files and to syslog server `loghost`:

```
node$ logger -p mail.crit "test message mail.crit"

node# cd /var/log ; tail -1 maillog ; tail -1 syslog
timestamp node username: test message kernel.warn
timestamp node username: test message kernel.warn

loghost# cd /var/log ; tail -1 maillog ; tail -1 syslog
timestamp loghost username: test message kernel.warn
timestamp loghost username: test message kernel.warn
```

---

4 System Backups

Backups are essential to continuity-of-service. Backups can help restore a system following a hardware failure or a user (or system administrator) error.

Utilizing backups to recover from a security break-in event, however, is not the best means to recover. While the break-in was noticed now, it may not be immediately apparent when the actual break-in occurred – in other words, how many backups ago are also compromised? The best reaction to a break-in event is to completely rebuild the system from the ground up. In this manner, the system starts from a known good state. However, there still is a vulnerability – in rebuilding the system from the ground up, the same vulnerability which was taken advantage of the first time has very likely been re-introduced. If the previously deployed version of FTP provided the vulnerability, rebuilding the system to the previous state will reintroduce that same vulnerability again. This is a paradox in that one needs to restore the system to service, but also remedy the vulnerability which caused the service disruption on the first place.

If the installation and rebuild procedures are well documented and developed, the core Operating System and configuration almost trivializes the rebuild process, meaning the system can be treated more as a "fuse" which can be tripped when a compromise occurs. The system disks could be rebuilt, or outright replaced. Being able to pull out the Operating System drive(s) and replace them with a known good copy allows for quick restoration of service. When the application data (user home directories, web pages…) are stored on physically different drives, it becomes simple to get the new OS disk(s) to return service with the full set of application data. Replacing the OS disk(s) is even an easier task with removable drives available in many current-day servers. By replacing the failed OS disk(s) with a spare, one cuts through the paradox of returning the system to service versus investigating the root-cause of the problem by 1) quickly returning the service with the spare and 2) retaining the disk image of the problem system for off-line investigation.

4.1.1 System Backups

If the method of treating the OS disks as "fuses" is utilized, it can eliminate the need for backups of the Operation System partitions. While this is the ideal, when reality steps in, backups of the OS partitions may be necessary. Due to the special files in the `/dev` directory, the `tar` and `cpio` utilities are not capable of providing full backup and restore capability to the OS partitions. For OS partition backups, `dump` should be utilized.

4.1.2 Data Backups

With the treatment of OS disks as "fuses", this leaves just the backup of the data areas: users home directories, application data, web pages, database content, etc.

Data backup solutions range from the simple to complex, and can accommodate small to vast data quantities. There are many commercial utilities on the market that provide Backup and Restore capability, from locally attached storage or through a backup network to a remote tape host or tape silo.

4.2 Local Host-based Data Backup Options

The utilities `tar` and `dump` could be employed to conduct backups of a small amounts of data on a local system. `tar` is a simpler utility than `dump` from a data backup point of view: `tar` is limited in that it cannot directly be used for incremental backups as `dump` can, and `tar` cannot be used to backup special files like those in `/dev`. `tar` could be used with incremental backups, but would require a support script to determine which files should be backed up as part of that incremental backup; `dump` can handle this automatically.

4.2.1 Using `dump`

An incremental backup is a backup of data since the last time a backup has been conducted. `dump` provides for 10 incremental levels: a "level 0" backup is a full backup, a "level 1" backup will include everything updated since the last level 0 backup, a "level 2" backup will include everything since the last level 1 backup (or if no level 1 backup, then since the last level 0 backup), and so on for successive backup levels. With a "level 0" backup conducted over the weekend, a "level 3" backup conducted on Monday, and a "level 4" backup conducted on Tuesday, the "level 5" backup on Wednesday will include everything that has changed since Tuesday's backup. The timestamps of the last dumps are recorded in `/etc/dumpdates`. If an incremental backup is conducted with no previous backups in `/etc/dumpdates`, that incremental is effectively a "level 0" backup.

Basic* `dump` *Usage

```
# dump [-0123456789u] [-f file] filesystem
```

`-u`	update `/etc/dumpdates` with the timestamp of this dump. this is used for determining the start time for incremental backups.
`-n`	where *n* is a number 0 through 9: the dump level
`-f file`	to specify the file to write to: this can be a file (e.g. /var/tmp/dump.dmp) or a raw device such as a specific backup disk or tape drive
filesystem	the file system to dump. This can be specified by the mount point or by the `/dev` filesystem name.

Below is a sample backup scheme that can be used as a starting point for a local backup scheme. In it, everything is backed up on the weekend, and incremental backups are conducted on the users home directories every evening shortly following the close of business.

system partitions for dump example:

```
/boot
/
/var
/usr
/home
/backup        (strictly for capturing the backups)
```

Saturday, 1 AM

```
# dump 0fu /backup/boot.dmp.0 /boot
# dump 0fu /backup/root.dmp.0 /
# dump 0fu /backup/var.dmp.0  /var
# dump 0fu /backup/usr.dmp.0  /usr
# dump 0fu /backup/home.dmp.0 /home
```

Monday, 6 PM

```
# dump 3fu /backup/home.dmp.3 /home
```

Tuesday, 6 PM

```
# dump 4fu /backup/home.dmp.4 /home
```

Wednesday, 6 PM

```
# dump 5fu /backup/home.dmp.5 /home
```

Thursday, 6 PM

```
# dump 6fu /backup/home.dmp.6 /home
```

Friday, 6 PM

```
# dump 7fu /backup/home.dmp.7 /home
```

***Adding Compression to* `dump`**

To add compression to the dump command sequences above, change the command to:

```
# dump 0fu - /boot | gzip -9fc > /backup/boot.dmp.0.gz
```

`dump` options explanation:

- `-f -` argument in place of the file name means redirect the file output to standard out.

`gzip` options explanation:

- `-9` maximum compression.
- `-f` force compression, even if there is no compression can be achieved (without this forced compression, some dumps may not be compressed and left in raw dump format, causing problems when restoring if the dump is expected in compressed format)
- `-c` write output to standard out.

4.2.2 Using `restore`

The complementary action to `dump` is `restore`. A backup is worthless if one cannot or does not know how to restore from it.

`restore` can be used to restore the entire contents of a `dump` backup, or be used to individually select portions of a backup to restore.

If a dump file was compressed as above, uncompress it before initiating the restore:

```
# cat x.dmp.gz | gunzip -c | restore -f - <other-restore-options>
```

4.2.2.1 Interactive Restore

It is a good safety practice to conduct interactive restores in a temporary working space, confirm these are the correct contents, and then move them into their proper location.

```
# cd /var/tmp
# mkdir restore
# cd restore
# restore -if x.dmp
```

- `-i` for interactive restore
- `-f` for restore from a file (the same as with dump)

```
restore >
```

Some of the options in interactive restore are:

`help`	list interactive restore command summary.
`quit`	quit restore.
`what`	list dump header information.
`ls`	list the contents of the working directory within the backup. Files and directories currently on the extract list will be denoted with an asterisk (*).
`cd`	change backup working directory.
`add` *what*	add file or directory to the extract list.
`delete` *what*	remove file or directory from the extract list.
`extract`	extract the files on the extract list.

```
restore > extract
Specify next volume #: 1
set owner/mode for '.'? [yn] y
restore >
```

Dump volumes are for dumps on removable media (e.g. tapes), and with a disk dump image, the only volume is the first one.

4.2.2.2 Full Restore

To restore the entire backup contents, a restore option other than "-i" is used:

```
# cd /var/tmp
# mkdir restore
# cd restore
# restore -rf x.dmp
```

This will restore the contents of the entire `x.dmp` file to the current directory (`/var/tmp/restore`, in this example).

4.2.2.3 Incremental Restore

To restore from incremental backups, one needs to search through the several incremental volumes in reverse order to find the most recent edition of the file. Using the daily incremental backup scheme outlined above, if the file targeted for restore hasn't been updated since Monday afternoon, and the current day is Thursday, search through the Wed and Tue backups and not find the file at all; it would be on the Monday backup, from which would complete the restore. If the file wasn't updated at all that week, after searching through the entire week of incrementals and not locating the file, fall back to the last full backup to complete the restore.

---

5 Installation Development and Testing

5.1 Test Environment

In addition to the deployed systems, there should be a test environment consisting of network-isolated and physically secured systems on which to engineer the operating system configuration – security elements, patch updates, application testing, inter-operability testing, intrusion-detection-system master configuration files, and so on. In this test environment, engineer and certify the operating system builds and updates even before they reach the production environment. It is with this test environment discipline that one can manage such a large scale of systems: most of ones attention is to this test environment, working through the configurations and issues, researching the solution, then discovering and documenting it. Then when it is time to roll out the solution it practically becomes a non-event.

5.2 Justifying a Test Environment

The return on the investment of a test environment increases with the importance of the production environment. An e-commerce company running a web server farm shouldn't be doing a "make all; make install" of an Apache upgrade on their production web server; instead, that upgrade should be compiled, installed and fully tested in a test environment that matches as closely as possible to the real production environment. If there is a minor blip, customers may notice. If there's a major outage, customers will really notice.

Disciplined engineering just isn't developing the installation process, but also developing and testing the backout process: what if something not comprehended during the initial testing happens and that upgrade needs to be backed-out. One must have the ability to undo what was done to reach the previous known good state. Without this documented and tested backout step, risk is increased. How long does it take to rebuild the server? How much does it cost for an hour of downtime? The answers to such questions provide justification for this test environment.

5.3 What To Test

The system should be thoroughly tested prior to deployment to ensure it is configured as it is intended. It should be tested standalone, and in a test network environment with other systems. Testing should include penetration testing and denial of service testing. Was anything missed? Was anything broken?

The range of tests should include everything from the simple and trivial, to the complex.

Can one telnet/FTP/rlogin into the system remotely? (should one be able to?)
Can one telnet/FTP/rlogin into the system from localhost?
Can one SSH into the system?
Does NTP connect to the NTP server, receive updates and stay in synch?
Can one NFS mount directories that are supposed to be mountable?
Can one NFS mount directories that are not supposed to be mountable?
…

One should test everything that is configured and expected to be functional on the system: if the configuration supposed to disable the "VRFY" feature inside sendmail, test it to ensure that the goal has been accomplished. If root is not supposed to be able to log in through a remote SSH login session, try logging in as root through an SSH login session. Maybe the configuration file has been modified, but that change doesn't take effect until the system is rebooted. Maybe the configuration file which has been altered is a deprecated, no longer used configuration file. Case in point:

`/etc/ftpusers`. Many test cases for the configurations provided above have been provided along with that configuration.

Other test scenarios to explore: Try to break into the system. But that in and of itself may not be the best strategy to take. Rather, test the system against the configuration with which it is intended to be deployed, addressing the question of has the intended goal been accomplished. Compare this to the physical world: at home, one can test break-in strategies by conducting a smash-and-grab exercise: throw a rock through the window, take the television and be gone. However, protecting against someone tossing a rock through your window to gain entry – is that something you are really wanting to protect against? As a homeowner, would you be willing to deploy the measures necessary to protect against this entry method? Refer to Fred Cohen's article in the reference section for a thought-provoking article on the topic of testing a system by breaking into it.

On the flip side, if there is a tool out there (`nmap`, `nesseus`) which others may have at their disposal, one can use that same tool to tell what others will eventually discover about these systems anyway. If one knows about it now while developing the build, one can take measures to close the gap now.

5.4 Testing Tools

There are a variety of tools that can be used to quickly get a test suite going.

5.4.1 Configuration Testing

Throughout the security configuration discussions in Section 2.4, test cases are listed for many of the individual configurations. These test cases can be used as the starting point for configuration testing.

5. §2.4.1 a) – validate md5 password hashing enabled.
6. §2.4.1 b) – validate password expiration setting.
7. §2.4.1 c) – validate password minimum length setting.
8. §2.4.1 f) – validate strong password checking setting.
9. §2.4.2 a) – validate ctrl+alt+del rebooting disabled.
10. §2.4.2 c) – validate root cannot log in remotely.
11. §2.4.3.3 – validate `/sbin/noshell` logging.
12. §2.4.4.1 – validate `sudo` is active.
13. §2.4.5 a) & b) – validate `su` limited to users in wheel group.
14. §2.4.7 a) – validate telnet service message.
15. §2.4.7 c) – validate sendmail service message.
16. §2.4.7 d) – validate FTP service message.
17. §2.4.10.2 d) – validate sendmail HELP, EXPN, VRFY disabling.
18. §2.4.11 a) – validate TCP wrappers hosts.deny configuration.
19. §2.4.13 d) – validate NTP service.
20. §2.4.15.1 b) – validate iptables flushed.
21. §2.4.15.1 d) – validate iptables allow traffic on loopback interface.
22. §2.4.15.1 e) – validate iptables allow outgoing interface conversations.
23. §2.4.15.1 g) – validate iptables allow incoming DNS resolution.
24. §2.4.16 a) – validate filesystems mounted read-only.

5.4.2 System Scanning

Prior to any scanning, be sure authorization is granted. And get it in writing.

In the best scenario, one should scan only the development system on an isolated network. If such a network is not at hand, use a cross-over cable to directly connect the development system to the "scanning launch" node.

Remember that the purpose of conducting a scan is to highlight vulnerabilities in the current configuration. After the scan, results should be shared on a need-to-know basis only.

5.4.2.1 Scanning from the Inside Looking Out

`netstat -vatup`	port information.
`lsof`	port and process information.
`ps`	process information.

netstat

`netstat` can be used to provide a variety of information about the networking system: network connections, routing tables and interface statistics.

```
# netstat -vatup
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program Name
tcp        0      0 *:ftp           *:*               LISTEN   709/xinetd
tcp        0      0 *:ssh           *:*               LISTEN   620/sshd
tcp        0      0 *:telnet        *:*               LISTEN   831/xinetd
udp        0      0 localhost:ntp   *:*                        575/ntpd
udp        0      0 myhost:ntp      *:*                        575/ntpd
udp        0      0 *:ntp           *:*                        575/ntpd
```

`netstat` command option explanation:

- `-v` verbose.
- `-a` "all": both listening and non-listening sockets.
- `-t` list TCP based services.
- `-u` list UDP based services.
- `-p` list process id and program name for sockets.

The `netstat` output identifies what ports and services are open on the system, and what processes is associated with that port. On this system FTP, SSH, telnet and NTP services are running. Compare what is measured here with `netstat` to what is intended to be functioning on the system. If there is a difference between measured and expected, the configuration is not quite correct yet.

With a port number is listed instead of the text description (for example, *:999 instead of *:ftp), this means the port number is not known to the system. Port names are identified in `/etc/services`. It is a good practice that applications define their used ports in `/etc/services`, but that practice isn't always followed. While inspecting the system, if a port number is not recognized, investigate what that port really provides. The last column, PID and program name, should provide clues to start the investigation if that port being open is legitimate or not.

It is also possible that, on a compromised system, the `/etc/services` port number to name translation to have been altered. To validate, use the `-n` option with `netstat`, and compare the reported port number with the actual service port number. Also, utilize the host intrusion system (AIDE, Tripwire) to validate correct `/etc/services` content.

```
# netstat -vatupn
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program Name
tcp        0      0 0.0.0.0:21      0.0.0.0:*         LISTEN   709/xinetd
tcp        0      0 0.0.0.0:22      0.0.0.0:*         LISTEN   620/sshd
tcp        0      0 0.0.0.0:23      0.0.0.0:*         LISTEN   831/xinetd
udp        0      0 127.0.0.1:123   0.0.0.0:*                  575/ntpd
udp        0      0 10.5.5.45:123   0.0.0.0:*                  575/ntpd
```

```
udp        0      0 0.0.0.0:123     0.0.0.0:*                           575/ntpd
```

This measure will report the real number, in case `/etc/services` is written with bogus entries: just because http normally runs with port 80, a nefarious service could be hidden as http by appending an entry for http to another port in `/etc/services`. If the inspection only consists of verifying the name, not the actual port number, this alternate http service would go unnoticed.

lsof

`lsof` will "list open files" on the system. `lsof` is an excellent diagnostic tool with a great many uses. First, let's consider only the ports which are open on the system, so `lsof` output will be limited to files of IP nature.

```
# lsof -i -M
```

```
COMMAND     PID USER   FD   TYPE DEVICE SIZE NODE NAME
xinetd      831 root    3u  IPv4   1146       TCP *:telnet (LISTEN)
ntpd        904 root    4u  IPv4  97050       UDP *:ntp
ntpd        904 root    5u  IPv4  97051       UDP localhost:ntp
ntpd        904 root    6u  IPv4  97052       UDP l72b:ntp
sshd      27134 root    3u  IPv4  57811       TCP *:ssh (LISTEN)
```

`lsof` command option explanation:

- `-i` select limit display to "IP" files.
- `-M` don't display portmap registration (`+M` to display portmap registration)

This `lsof` output can be compared and contrasted to the `netstat` output.

ps

The `ps` command lists the process table. This can be used to analyze if processes are running which have been thought to have been disabled (with `chkconfig`, to remove from the run-control script). `ps` by itself is less chatty than `lsof` is, for investigating if a process exists.

```
# ps -fade
```

```
UID      PID  PPID  C STIME TTY          TIME CMD
root       1     0  0 Feb08 ?        00:00:04 init
root       2     1  0 Feb08 ?        00:00:00 [keventd]
root       3     0  0 Feb08 ?        00:00:00 [ksoftirqd_CPU0]
root       4     0  0 Feb08 ?        00:00:25 [kswapd]
root       5     0  0 Feb08 ?        00:00:00 [kreclaimd]
root       6     0  0 Feb08 ?        00:00:00 [bdflush]
root       7     0  0 Feb08 ?        00:00:00 [kupdated]
root       8     1  0 Feb08 ?        00:00:00 [mdrecoveryd]
root      12     1  0 Feb08 ?        00:00:00 [kjournald]
root     135     1  0 Feb08 ?        00:00:00 [kjournald]
root     136     1  0 Feb08 ?        00:00:00 [kjournald]
root     137     1  0 Feb08 ?        00:00:00 [kjournald]
root     546     1  0 Feb08 ?        00:00:00 syslogd -m 0
root     589     1  0 Feb08 ?        00:00:00 ftp: accepting connections on p
root     865     1  0 Feb08 ?        00:00:00 crond
daemon  1052     1  0 Feb08 ?        00:00:00 /usr/sbin/atd
root   17082     1  0 Mar20 tty1     00:00:00 /sbin/mingetty tty1
root   17134     1  0 Mar20 ?        00:00:00 /usr/sbin/sshd
root   17298     1  0 Mar20 tty4     00:00:00 /sbin/mingetty tty4
root   17300     1  0 Mar20 tty2     00:00:00 /sbin/mingetty tty2
root   17342     1  0 Mar20 tty3     00:00:00 /sbin/mingetty tty3
root   17385     1  0 Mar20 tty5     00:00:00 /sbin/mingetty tty5
ntp    18104     1  0 Mar29 ?        00:00:03 ntpd -U ntp
root   18665     1  0 Mar30 tty6     00:00:00 /sbin/mingetty tty6
root   10222 10221  0 09:09 pts/0    00:00:00 login -- xdevel
xdevel 10223 10222  0 09:09 pts/0    00:00:00 -bash
root   10264 10223  0 09:09 pts/0    00:00:00 su -
```

```
root   10265 10264  0 09:09 pts/0 00:00:00 -bash
root   10411 10265  0 09:32 pts/0 00:00:00 ps -fade
```

When considering the process table, there are several clues to help understand where a process might be initiating from.

TTY: terminal which initiated the process.
The pts/0 processes (near the bottom of the list) are from the current login session, connected over the pseudo-terminal.

PID: process id.
Process id's start at 0 and work there way up, recycling numbers after reaching roughly 32,000. System processes usually have lower PIDs and, if the process is permanent and doesn't recycle, will retain that PID for the life of the system. If system processes are restarted, they will likely have a higher PID and the STIME will change: if a process normally has a lower PID (say, under 200), and suddenly changes, this could indicate a problem.

PPID: parent process id.
The PPID is the process id of the process which initiated this process. This is helpful for tracing down where a process comes from. Most of the system level processes are initiated by init, pid=1. To assist tracking down the chain of processes, consider adding the `--forest` option to `ps`.

STIME: process start time.
If a process was started within the last 24 hours, start time will be reported as the hour and minute it was started; otherwise it will be reported as the date which it started. From the ps listing above, it can be concluded that the system was started on Feb 8th (all the system level processes started then). Someone last used the console on March 20th (the tty's processes were started then). The NTP daemon was restarted on March 29th.

5.4.2.2 Scanning from the Outside Looking In

nmap	port scanner; can attempt to fingerprint the OS brand name and release.
rpcinfo -p	portmap information.

When scanning the system over the network, be sure to do so over a private subnet. That is, hosts connected directly to each other without any connection to other systems (the Internet, or other systems at the site). Ideally, conduct a network scan between two directly connected systems. Under such a network hookup can one be absolutely certain that there will be influence or impact to other systems by the network scan.

nmap

`nmap` is a utility to scan a system over the network. It can be configured to scan a single system, or a multitude of systems. `nmap` can also be used an attack tool.

`nmap` can be used to conduct TCP, UDP and OS fingerprint scans.

nmap -vv *targethost*

Conducts a TCP scan of system *`targethost`*, and will list ports open on the remote system.

```
Port        State      Service
22/tcp      open       ssh
23/tcp      open       telnet
123/tcp     closed     ntp
```

`nmap` command options:

`-v` verbose output
`-vv` very verbose output

`-sT`	TCP port scan (default)
`-sU`	UDP port scan
`-O`	use OS fingerprinting to guess Operating System brand and release version
`-sP`	ping scan, find if machine/s are reachable
`-sR`	RPC scan
`-n / -R`	never/always resolve with DNS
`-p`	specify port/s to check

nmap -vv -sU -p 1-1024 *targethost*

Conducts a UDP scan of system *targethost*, and will list ports open on the remote system. A UDP port scan take a very long time, so it is wise to limit the range of scanned ports if the scan needs to be conducted quickly.

```
Port        State       Service
111/udp     open        sunrpc
123/udp     open        ntp
```

rpcinfo

The `rpcinfo` command lists all registered RPC programs, on the localhost or the host specified.

rpcinfo -p

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100024    1   udp   1024  status
    100024    1   tcp   1024  status
```

Investigate the source of each registered RPC program: if the system does not have any, it will report:

```
rpcinfo: can't contact portmapper: RPC: Report system error - Connection refused
```

The `rpcinfo` command can also be issued remotely:

rpcinfo -p *targethost*

If no host argument is provided, localhost is assumed.

---

6 System Deployment and Sustaining

6.1 Deployment Ready?

Only after testing has verified the intended configuration can the system be considered ready for deployment.

Document everything. The installation process should be repeatable and result in the same implementation. If it is not documented, the configuration may have to reinvented or reverse-engineered at a later time.

6.2 System Sameness

Once the implementation standard has been defined, ensure that the same implementation is uniformly deployed. A human-based installation method is prone to error (despite all the best intensions) and cannot scale to a large number of systems economically. A number of methods can be used to duplicate an installation:

- direct disk duplication (perhaps leveraging hardware disk mirroring).
- build an auto-install CD.
- build image deployment software, such as Norton's Ghost product.
- leverage KickStart for installing the base OS, and write scripts to install everything after that.

Whatever the solution employed, spend some time both preparing and testing it. Conduct the install per the method, and test that installation too. Even better, have someone else install per the method, and test that installation. There'd be nothing worse to have spent all the time and effort to develop a configuration to provide a certain functionality or protection, and then to have it omitted from the full deployment.

Only with an automated installation system can there be the guarantee of system sameness.

6.3 Monitoring

After the system and its support infrastructure has been deployed, security doesn't stop. The system and environment need to be monitored continuously.

Server logs, on the FTP server and syslog server, should be checked periodically. Even though careful examination of logs is too time-consuming to be realistic, the logs should be checked for suspicious behavior. The syslog output on the individual servers should be compared to the logs on the central syslog server: are there entries in the central server missing on the individual server? If so, this could be an indicator of a problem on the individual server.

Packet filter rules need to be tweaked as new applications requiring new ports come into the environment.

The system time needs to be double-checked periodically so verify they are truly keeping time synchronized.

6.4 Patching

And the systems also need to stay current with Operating System and application patches. Updates include several steps:

1. _____ apply the patch within the test environment
   See if it fixes what it's supposed to fix. If possible, acquire or develop a test case to demonstrate the condition that is being fixed. Demonstrate the problem in the current environment, install the patch, then demonstrate that the problem has been fixed. This test

case should be retained and used again when testing future updates to validate that a problem previously fixed is not reintroduced.

2. _____ develop install and backout instructions
   Test the install and backout procedures.
3. _____ compatibility test the patch
   See that it doesn't break anything else.
4. _____ deploy the patch throughout the environment
   Maintain system sameness.
5. _____ update the system integrity checker database (e.g. AIDE, Tripwire)

The first three steps are all conducted in the test environment. With the third step, testing is conducted with other elements of the Operating System, but also with applications. If a patch does address a vulnerability but doesn't play nicely with everything else (other OS elements, and/or other applications) on the system, you then have a difficult decision to entertain: do you deploy a fix that breaks something else?

6.5 Patch Deployment

After it has been demonstrated that the update can be both applied and removed, that the patch itself functions as advertised, and that it does not break anything else, it can be deployed to the entire environment. Distribute the patch across the entire infrastructure, and apply the patch uniformly to maintain system sameness. Don't forget to update the integrity checker database, otherwise the update will generate lots of false alerts.

6.6 The Journey Continues

System security is not a walk in the park; it's a safari through the jungle, and there are lots of lions, tigers and bears (*oh my!!*) in that jungle…and lion traps, tiger traps, and bear traps. The steps and methods presented here only begin the journey – this document only scratches the surface. There's always more that could and should be done. Entire books have been written about some topics, there's no ability to do such topics justice in such a short presentation as this.

Plan for Failure

Murphy's Law says "anything that can go wrong will go wrong". Plan for failure. Anticipate an incident. What can be done now to prepare for it? To mitigate it? Or make it not possible to affect these systems? Even if nothing can be done to adequately prepare for it, when the incident happens one will know how to react and recover.

---

7 References

7.1 Sources

Anonymous; *Maximum Linux Security*; SAMS Publishing, 2000.

Brotzman, Lee E.; "Linux Security: Step-by-Step"; *SANS Institute, Securing UNIX Systems, 6.5, Linux/Solaris Practicum*; Oct. 2001.

Brotzman, Lee E., and David A. Ranch, eds.; *Securing Linux Step-by-Step, version 1.0*; SANS Institute, 2000.

Brotzman, Lee E., and Hall Pomeranz; "Running UNIX Applications Securely"; *SANS Institute, Securing UNIX Systems, 6.4, Running UNIX Applications Securely*; Oct. 2001.

Brumley, David; "Repeatable Security"; *;login*; Nov 2000; vol. 25, no. 7; pp 32-37.

Burgiss, Hal; *Security Quick-Start HOWTO for Red Hat Linux*; v1.1d, 17-Dec-2001; from http://www.linuxsecurity.com/resource_files/documentation/QUICKSTART-Redhat/.

Cohen, Fred; *Managing Network Security: Testing Your Security by Breaking In – NOT*; Strategic Security Intelligence; from: http://www.all.net/journal/netsec/0102.html.

Drake, Joshua; "10 Minutes to an iptables-based Linux Firewall"; *LinuxWorld*; from: http://www.linuxworld.com/site-stories/2001/0920.ipchains.html; 20-Sep-2001.

Lehti, Rami, and Pablo Virolainen; "AIDE (Advanced Intrusion Detection Environment)"; from http://www.cs.tut.fi/~rammer/aide.html.

Lundberg, Gregory A; WU-FTPD Upload Configuration HOW-TO; from: http://www.wuftpd.org/HOWTO/upload.configuration.HOWTO.

Napier, Duncan; *IPTables/NetFilter – Linux's Next-Generation Stateful Packet Filter*; SysAdmin, 2001; from: http://www.samag.com/documents/s=1769/sam0112a/0112a.htm.

Pomeranz, Hal; "UNIX Security Tools"; *SANS Institute, Securing UINX Systems, 6.2, UNIX Security Tools*; Oct. 2001.

Toxen, Bob; *Real World Linux Security: Intrusion Prevention, Detection and Recovery*; Prentice-Hall PTR, 2001.

van den Hout, Koos; *Frequently Asked Questions about wu-fptd, with answers*; from: http://www.wuftpd.org/wu-ftpd-faq.html.

http://www.xinetd.org/; xinetd.

7.2 For Further Reading

AIDE

http://www.cs.tut.fi/~rammer/aide.html

Packet Firewalls

http://www.linux-firewall-tools.com/linux/

http://www.linuxdoc.org/LDP/nag2/x-087-2-firewall.html detailed comparison of ipchains and iptables

Packet Firewalls – iptables

http://www.linuxsecurity.com/resource_files/firewalls/IPTables-Tutorial/iptables-tutorial/iptables-tutorial.html
http://www.linuxworld.com/site-stories/2001/0920.ipchains.html
http://netfilter.samba.org/unreliable-guides/packet-filtering-HOWTO/

Linux Security

http://www.linuxsecurity.com/

Linux Updates, Red Hat

http://redhat.com/apps/support/errata/ Red Hat's errata notices.
https://listman.redhat.com/mailman/listinfo/redhat-watch-list Mailing "watch list" for security.
http://www.redhat.com/download/mirror.html Update distribution mirror list.

Nessus Network-based vulnerability assessment and reporting tool.
http://www.nessus.org/

NMAP Port scanner.
http://www.insecure.org/nmap/index.html

NTP Network Time Protocol, distributed with Red Hat Linux 7.2.
http://www.ntp.org/
http://www.eecis.udel.edu/~mills/ntp/servers.htm

TARA Host-based vulnerability assessment tool.
http://www-arc.com/tara/index.shtml

Tripwire Host-based integrity checking (commercial tool).
http://www.tripwire.com/
http://www.tripwire.org/ Open Source version of the commercial tool

SARA Network-based vulnerability assessment and reporting tool.
http://www-arc.com/sara/

Snort Network intrusion detection tool.
http://www.snort.org

xinetd eXtended InterNET services daemon, distributed with Red Hat Linux 7.2.
http://www.xinetd.org/
http://www.linuxfocus.org/English/November2000/article175.shtml

---

8 Appendix

8.1 Programs

8.1.1 noshell.c

```
#include <stdio.h>
#include <syslog.h>
#include <unistd.h>

int main()
{
  printf("access denied.");
  openlog("noshell", LOG_NDELAY, LOG_AUTHPRIV);
  syslog(LOG_ERR, "intruder alert, noshell account '%s' accessed.",
    getlogin());
  closelog();
  exit(0);
}
```

8.1.2 iptables_config

```
#!/bin/sh

# iptables_config
# chkconfig: 2345 09 91
# description: configure local packet-filter with iptables

PATH=/sbin:/bin

# restrict connections to my network
MYNET=169.254.1.1/16
# allow connections from anywhere
#MYNET=0/0

ALLOWSERVICES="20/tcp 21/tcp 23/tcp 123/udp 514/udp 137:139/both"

#####

echo " ... flusing existing tables"
iptables --flush
iptables --flush -t nat

echo " ... closing all traffic"
iptables --policy INPUT   DROP
iptables --policy OUTPUT  DROP
iptables --policy FORWARD DROP

echo " ... open all traffic on loopback interface"
iptables -A OUTPUT -j ACCEPT -o lo
iptables -A INPUT  -j ACCEPT -i lo

echo " ... allow all outgoing conversations"
ST_NER="--state NEW,ESTABLISHED,RELATED"
iptables -A OUTPUT -p tcp -m state $ST_NER -j ACCEPT
iptables -A OUTPUT -p udp -m state $ST_NER -j ACCEPT

echo " ... allow some ICMP"
# this configuration will allow out-going pings,
# but will not respond to incoming pings.
```

```
iptables -A INPUT -p icmp --icmp-type echo-reply \
      -s $MYNET -j ACCEPT
iptables -A INPUT -p icmp --icmp-type destination-unreachable \
      -s $MYNET -j ACCEPT
iptables -A INPUT -p icmp --icmp-type time-exceeded \
      -s $MYNET -j ACCEPT

echo " ... allow incoming DNS resolution"
iptables -A INPUT  -p udp -m state $ST_NER --source $MYNET \
      --sport 53 -j ACCEPT

echo " ... opening services:\c"
for service in $ALLOWSERVICES; do
  port=`echo $service | awk -F/ '{print $1}'`
  type=`echo $service | awk -F/ '{print $2}'`
  echo " $service\c"
  case "$type" in
    both)
      iptables -A INPUT  -p tcp -m state $ST_NER --source $MYNET \
          --dport $port -j ACCEPT
      iptables -A INPUT  -p udp -m state $ST_NER --source $MYNET \
          --dport $port -j ACCEPT
      ;;
    tcp)
      iptables -A INPUT  -p tcp -m state $ST_NER --source $MYNET \
          --dport $port -j ACCEPT
      ;;
    udp)
      iptables -A INPUT  -p udp -m state $ST_NER --source $MYNET \
          --dport $port -j ACCEPT
      ;;
    *)
      echo "improper specification '$service'"
      ;;
  esac
done
echo ""

iptables -A INPUT -j LOG -m limit --limit 20/minute \
      --log-prefix "iptables drop: "

#--end--
```

8.1.3 ftpd

```
#!/bin/sh

# ftpd
# chkconfig: 2345 35 35
# description: run ftp as stand-alone daemon

FTPPID=/var/run/ftpd.pid

PATH=/usr/sbin:/bin

getftppid() {
  ftpPID=`ps -fade | grep " ftpd: " | grep -v grep | awk '{print $2}'`
}

start() {
  echo -n "Starting ftp daemon: "
```

```
  in.ftpd -S -l -a -t 600 -w
  sleep 1
  getftppid
  if [ "_$ftpPID" = "_" ]; then
    echo -n "[none]"
    rm -f $FTPPID
  else
    echo -n "[$ftpPID] "
    echo $ftpPID > $FTPPID
    chmod 0644 $FTPPID
  fi
  echo ""
}

stop() {
  echo -n $"Stopping ftp daemon: "
  getftppid
  if [ "_$ftpPID" = "_" ]; then
    echo -n $"[none]"
  else
    echo -n $"[$ftpPID] "
    kill $ftpPID
  fi
  echo ""
}

case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  restart|reload)
    stop
    start
    ;;
  *)
    echo "Usage: $0 { start | stop | restart }"
    ;;
esac

exit 0

#--end--
```
