Global Information Assurance Certification Paper

Linux Repository Server: Implementing and Hardening Step by Step

Alexandre Lima de Abreu Teixeira
RHCE CCNA LPIC-2

SANS GCUX - Version 3.0
Option 1 - Securely Administering UNIX

December 2004

Content

Abstract

One of the highly critical roles in computers security maintenance is patch management, this paper discusses the process of implementing softwares and measures in order to successfully accomplish such role.

The main idea is to setup a web server on local network providing RPM[1] repository structure for Linux[2] clients, thus resulting in less bandwidth consumption and rapidly software updates. The content is focused on Fedora[3] Linux operating system, but it will be useful for users of any Linux distribution based on RPM package management system.

By using commonly deployed solutions such as Apache web server and Fedora distribution tools, this document will explain how to easily implement Linux patch management scheme. All steps needed to build and secure network services as well as the operating system will be covered in details.

This paper is intended to be a good reference for everyone who needs to maintain several Linux machines with its software and patches updated by using a simple and uniform way.

1 The RPM Package Manager (RPM) is a powerful command line driven package management system capable of installing, uninstalling, verifying, querying, and updating computer software packages. URL: http://www.rpm.org (10 Jul. 2004).

2 Linux is a free Unix-type operating system originally created by Linus Torvalds with the assistance of developers around the world. URL: http://www.linux.org (10 Jul. 2004).

3 The Fedora Project is a Red-Hat-sponsored and community-supported open source project. URL: http://fedora.redhat.com (10 Jul. 2004).

The Environment

GIAC Enterprises is a company that provides cargo transportation and logistics management as principal activities. The company's network currently has several Linux boxes including other Unix based operating systems for running critical mission applications and servers such as web applications and database management systems.

The network administrator has realized that there are many Linux machines to maintain and there are no technical skills and time to accomplish the task of patch management among those machines. The company then decided to hire an Unix administrator, he will be responsible for maintaining Unix and Linux boxes with focus on implementing an automated Linux patch management solution.

The following procedures will ensure that the target machine, the repository server, will be ready and secure for providing network services helping Unix administrator to manage and easily deploy patches for the internal Linux boxes.

GIAC Enterprises has its Internal network designed by using reserved IP addresses[4] with 172.16.0.0/16 network IP address, all users must configure their applications settings so that Internet browsing is done by using a proxy server.

The company has two links to outside world, one leased line used by Internal network users and an ADSL[5] link connected to a testing environment which network IP address is 192.168.1.0/24, as showed on Figure 1:

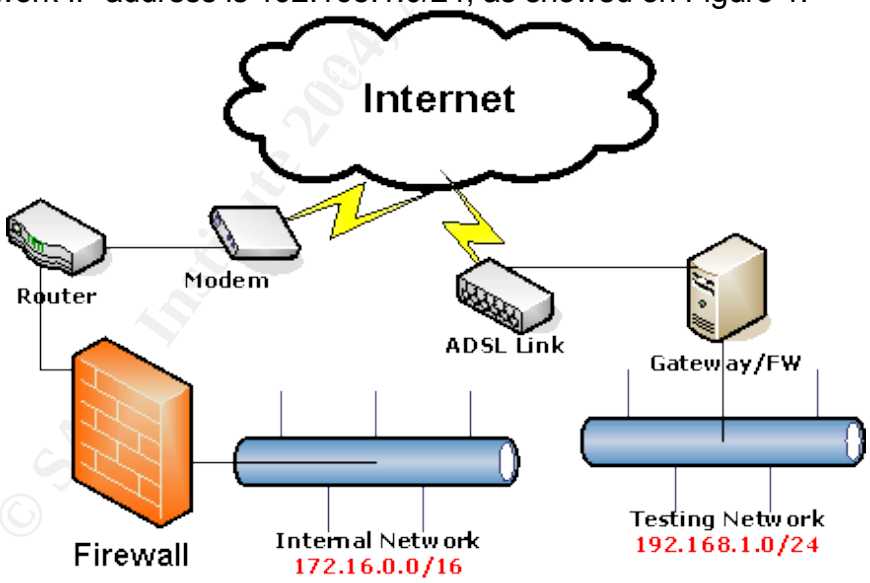

Figure 1 – Network Design of GIAC Enterprises

4 RFC Document "Address Allocation for Private Internets". February 1996. URL: http://www.rfc-editor.org/rfc/rfc1918.txt (12 Aug. 2004).

5 Short for ***a**symmetric **d**igital **s**ubscriber **l**ine.* URL: http://www.webopedia.com/TERM/A/ADSL.html (12 Aug. 2004).

Current Patch Management Scheme

The recently hired Unix administrator has realized that there are no rules when Linux users need to update and patch their systems. Figure 2 shows how this is happening: same packages from different locations, untrusted packages and less then a half of total security patches available are applied.

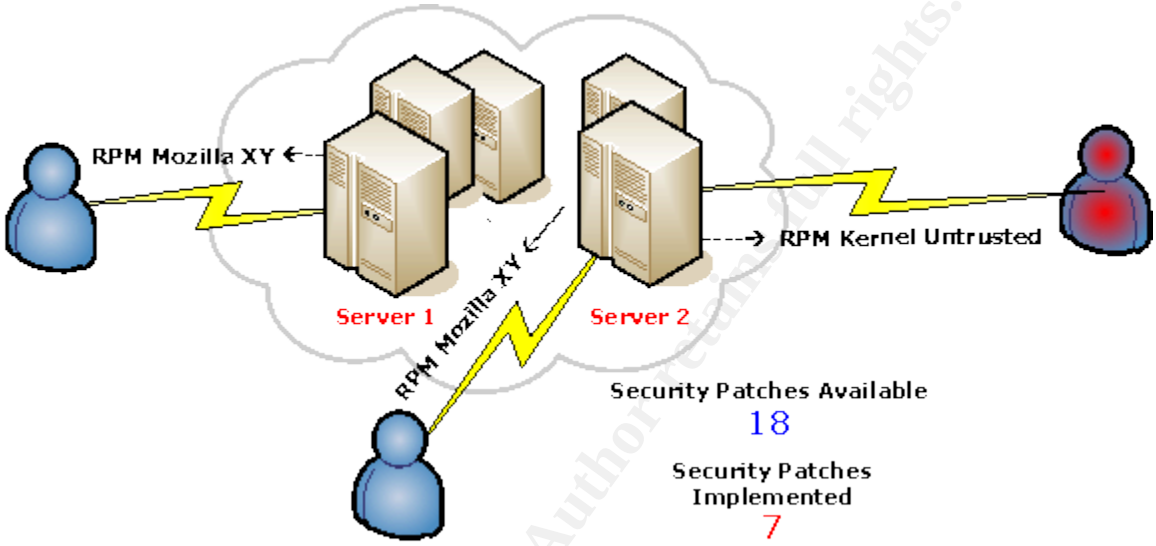

Figure 2 – Chaotic Patch Management

Packages are available on Internet, through some FTP/HTTP server mirror, which provides Fedora Project files, users who need patching compromised software just have to select one of those mirror servers and then download and install the specified update package. The problem is that every single package will be fetched by every user, resulting in many unnecessary resources consumption and other issues as listed below:

- Users hard disk space consumption

  There is no need to store the same package in many machines, storing the package at only one site will reduce disk utilization;

- Higher Link bandwidth utilization

  Users may choose different mirrors, thus the same package will be downloaded more than once, this usually happens when a critical patch becomes available on Internet;

- There is no packages integrity and validity verification procedures, so users may download packages from untrusted sources;

- It's almost impossible to reach an uniform package level application among the machines, some users may have no patches installed.

Proposed New Model: Local and Centralized

Iuri, the Unix administrator of GIAC Enterprises, has the idea of building a local packages repository, where every needed packages will be downloaded and stored. The repository server will have its list of packages synchronized with a mirror server located outside and it will deliver packages download for local Linux users through HTTP protocol.

This new model will provide less outside connections when the process of updating Linux software begins since RPM packages will reside on local repository server, providing Linux users a single and locally connected server to point to when software update is needed.

Figure 3 shows how the local repository will work and illustrates some advantages of the new proposed model:

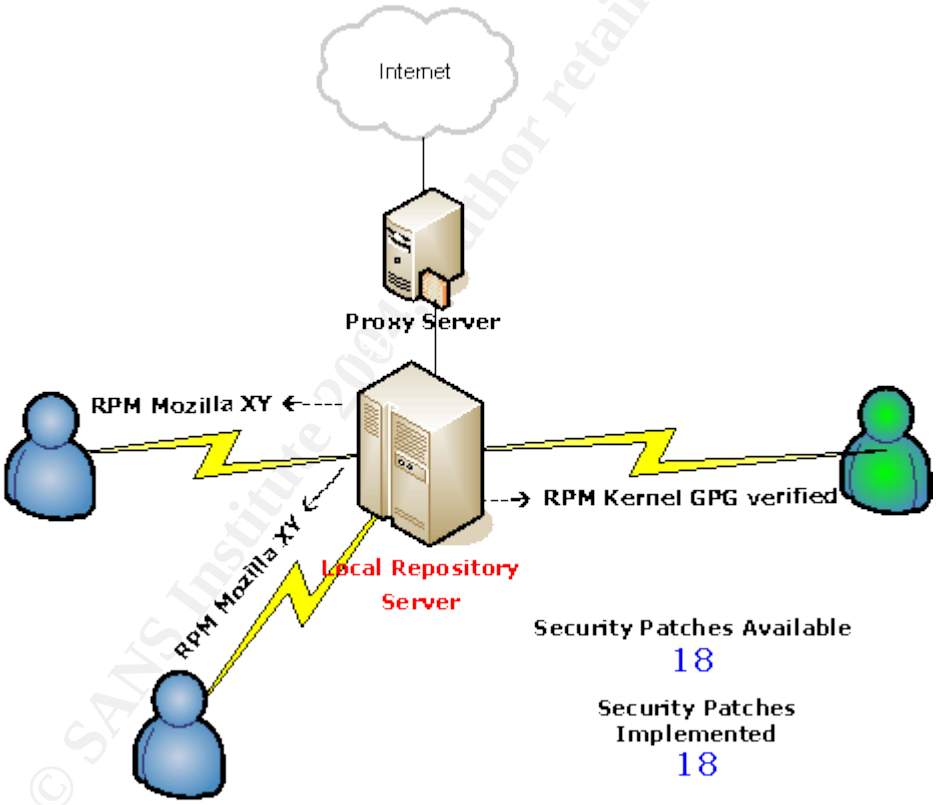

Figure 3 – Iuri's Linux Patch Management Scheme

Iuri has decided to place the new server on Internal network, it will use the existing local proxy server for downloading RPM packages from Internet. The complete action plan includes all steps necessary for mitigating all security issues generated by his decisions.

Hardware and Software Description

The main purpose of the repository server will be packages availability through Internal network, so there's no need of high speed processor but enough storage space. Iuri has decided to implement the repository server using the hardware described below:

Hardware	
Processor	AMD Duron 800mhz
Storage	IDE 80GB
Memory	512MB
Network	10/100 Ethernet LAN

The repository server will be running with Fedora Core Linux version 2, Apache as the web server and OpenSSH for remote server administration and shell access.

One essential software required for implementing the solution is Yum: Yellow dog Updater, Modified. Yum is an automatic updater and package installer/remover for rpm, it automatically computes dependencies and figures out what things should occur to install packages. It makes it easier to maintain groups of machines without having to manually update each one using rpm[6]. Wget program will be used to download packages from Internet and Logrotate program will be used to rotate server's logs.

Software	
Operating System	Fedora Linux 2
Network Services	Apache 2.x, OpenSSH 3.6.1p2+
Custom Script	yum_repository.sh[7]
Additional Required	Yum 2.0.7+, Wget 1.9.1+, Logrotate 3.7+

Iuri will need to install a customized shell script called *yum_repository.sh*, this will run as a *cron job* on repository server, it will be responsible for running *wget* and *yum-arch* programs for packages downloading and repository tree building, respectively.

6 URL: http://www.linux.duke.edu/projects/yum/ (29 Aug. 2004).
7 URL: http://fedoranews.org/alex/scripts/yum_repository.sh (29 Aug. 2004).

Some additional packages required for correct installing and maintaining the solution will be also installed, these steps will be covered in details later.

The complete list of software used is listed on Appendix A. Source code of all Custom Scripts and Configuration files are described on Appendix B.

Clients Standpoint Requirements

Linux users must have installed on their machines some RPM based distribution with Yum package management program, its client will be responsible for RPM packages download from local repository server, packages dependencies calculation and user interaction between RPM base.

On GIAC Enterprises all Linux clients are based on the second version of Fedora Linux distribution where Yum program is found on distribution ISO images[8] as well as all software required for client side implementation.

8 URL: http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/iso/ (12 Sep. 2004).

New Model Effects and Security Issues

When Iuri was planning to implement the solution, he becomes aware about the security risks of introducing the new server to the existing environment. The following checklist needs to be reviewed before deploying the repository server on production environment:

1) Software Integrity Verification

All software used on the solution will need to be verified and checked against RPM tools and checksum tools. This includes Fedora distribution ISO images and all packages downloaded from Internet.

On Fedora Linux, Iuri will install public GPG key of package creator such as Fedora development team, so that every package will be verified against that key and also ISO images will be checked against a list of md5 checksums included on download site.

Security Risk: This measure will avoid users from installing and using untrusted softwares and it will ensure that operating system installation files are not corrupted.

2) Operating System Security

Before deploying the server on production environment, the server's tasks must be clearly defined. Iuri will apply all security patches available and a list of security enforcement items as part of operating system hardening process.

This process is also called “Baseline Strategy”[9], the Baseline document is a group of security settings that are designed for each type of computer on GIAC Enterprises. These definitions are prepared in such a way that the computer performs it functions or tasks, but nothing else.

Security Risk: The above steps implementation will minimize the risks of server local exploitation by an attacker such as local bug exploitation and privilege escalation by exploring operating system default settings and configurations.

3) Network Services Security

As it will be done for operating system, every network service installed on the server will be hardened too, this will make significant enhancements on security level of the system by avoiding remote bugs exploration.

Security Risk: Remote Exploitation by exploring a network service bug or possible incorrect configuration on a running service or daemon.

9 Koconis and co-workers, p.4.

4) Operating System and Services Auditing

After security baselines application and hardening on both the operating system and network services, Iuri will run some auditing tools against the server for validating operating system and network services security implementation.

Security Risk: By running auditing software against the server, most of known vulnerabilities and common bugs will be discovered, that will reduce the risks of a successful exploitation by an attacker since the administrator will take measures for eliminating all possible vulnerabilities and problems reported after the auditing process.

5) Access Policy Implementation

Iuri will setup network services for allowing only authorized users access. The server will be running OpenSSH daemon for remote access to Linux shell, so only defined users will have rights to log on. This will be based on some specific service settings and rules. The same method will be applied for web server access too.

The server will need Internet connection and other types of network access, only defined access will be allowed, this will be done on the server by implementing firewall rules.

Security Risk: Access policy will avoid services utilization by unknown or unauthorized users and will make sure that server will only have access to certain hosts and network services.

6) Homologation Directives Application

When beginning implementation, the procedures will be done in a testing environment, this will mitigate security risks of network attacks and will provide a good time window for services and network connections tests.

All implementation steps will be documented and then installed *from scratch* on production environment, any future or necessary changes must be validated on testing environment first and then deployed on final target.

Security Risk: Any future changes on the server have to be made on a testing environment to mitigate chances of server problems, which include network services availability.

To implement this action, users must have the necessary time to test and validate the solution and changes, the documentation is also important since the task of re-implementing or changing the environment could be delegated to another person.

7) Logging Enable Implementation

To increase possibly auditing and forensics application, the server will enable specific logging information about the server's operating system and services. This measure will be useful in cases of some server problem or crashing and other possible security incidents such as Denial of Services and Intrusion attempts.

Security Risk: Insufficient server data and log information in case of a server problem as described above.

8) Physical Security

Iuri will deploy the server on GIAC Enterprises data center, which will include additional security procedures and physical security improvements as listed below:

- Monitored air quality level measurement system installation
- Monitored physical access, including unique identification and accouting policy for every people that enters machines room
- Monitored fire detectors integrated with a system that avoid fire propagation not affecting hardware integrity
- Video cameras system installation for people entrance recording
- Backup power systems implementation
- BIOS password setup
- Portable devices removal after deployment

Security Risk: Physical unauthorized access to the server, server unavailability and lack of control about who enters servers room. Possibly software loading using removable medias(floppy disk and cdrom devices).

Implementation

Starting from Installation process until server deployment on production environment, all the following steps and configurations will be done on testing environment and documented by GIAC Enterprises Unix administrator, that is a requirement for mitigating security risks clearly described on checklist item "Homologation Directives Application".

Also note that only highly technical steps will be described in-depth, focusing on what this paper title proposes and in accordance with this document limitations and definitions.

Software Download and Preparation

In order to begin Linux installation, Iuri starts the process by downloading Fedora Core 2 ISO images from Fedora Project website. The files will be stored at some Linux machine located on test environment. The complete URL for the images are listed below:

http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/iso/FC2-i386-disc1.iso
http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/iso/FC2-i386-disc2.iso
http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/iso/FC2-i386-disc3.iso
http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/iso/FC2-i386-disc4.iso

After downloading the images, Iuri verifies the integrity of files by running *md5sum* Linux command, which is part of *coreutils* RPM package. That command returns the checksum of given files. The following URL shows the list of correct md5 checksums:

http://download.fedora.redhat.com/pub/fedora/linux/core/2/i386/iso/MD5SUM

This procedure will be in accordance with checklist item "Software Integrity Verification". The following information table illustrates the output of *md5sum* command to be executed in order to extract checksums of downloaded ISO images:

```
$ md5sum FC2-i386-disc{1,2,3,4}.iso
c366d585853768283dac6cdcefcd3a2d  FC2-i386-disc1.iso
fc3c926442cc85a469268651bd04c186  FC2-i386-disc2.iso
5ad870e696953f4bbd0a91936873890e  FC2-i386-disc3.iso
c736f8048b12315b5c0b070de1d74867  FC2-i386-disc4.iso
```

If any problem occurs when downloading the images, the verification will fail since the returned checksums will not match. As showed on output above, Iuri has downloaded the files correctly, every returned checksum matches original one.

Iuri then needs to record ISO images into CD medias to begin installation of operating system, this will require a CD recorder device. The medias can be recorded by using many Linux tools as described on the following document:

http://redhat.com/docs/manuals/linux/RHL-9-Manual/getting-started-guide/s1-disks-cdrw.html

This guide describes several ways of burning files and ISO images into writable CD medias using common Linux tools.

Iuri has chosen to use the CD Creator Interface that is integrated with *Nautilus* user environment, the package is called *nautilus-cd-burner*. To write ISO image into a CD, Iuri just had to right-click the image file with the mouse and then chose “Write to CD...” option as showed on Figure 4 below:

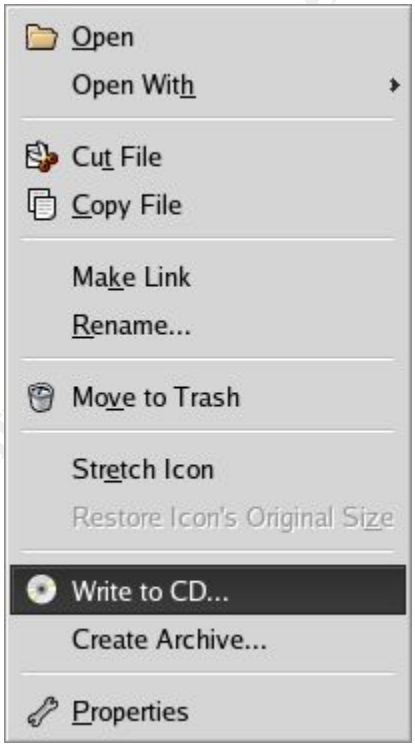

Figure 4 – Nautilus option for burning ISO image files after mouse click

This action has to be done for each one of the four ISO images, no more CD medias will be used instead of these. Next Installation step is about booting, installing and configuring the operating system software and packages.

Operating System Installation

At this point, after hardware tests, the machine designed for acting as GIAC Enterprises repository server will be used for installing Fedora Core 2 Linux distribution.

Some steps described on this section will be in accordance with checklist item "Operating System Security" such as Boot Loader password setup, which will be explained in detail later.

The following steps describe significant points that Iuri has taken in order to install Fedora Linux on repository server machine, there is no need to link the ethernet card on the network at this stage.

Fedora Core 2 Installation Steps

1. Insert the first media into the drive and power up the system;
2. Configure BIOS Setup in order to boot from CD ROM device and exit by saving settings, wait until system boot;
3. When the initial screen appears, type "linux text" for Text type Installation, as showed on Figure 5:

Figure 5 – Fedora Core 2 Text Installation

4. Next screen will ask about media checking, choose OK. If media check fails, try cleaning the disk first, if the media problem continues, try recording another media until the test runs OK. Figure 6 illustrates that installation stage:

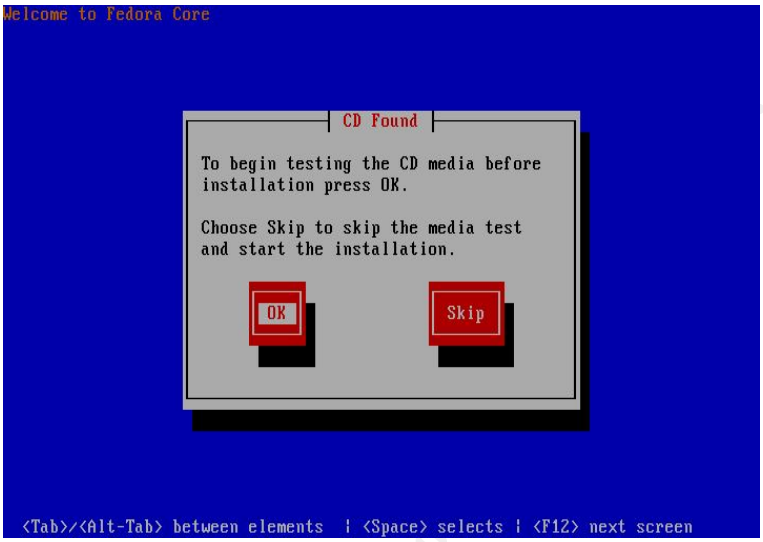

Figure 6 – Media Check Option

5. After a Welcome screen, the installer shows Language, Keyboard and Monitor selection screens, choose the right configuration for machine hardware and choose “Custom” Installation type, as showed below:

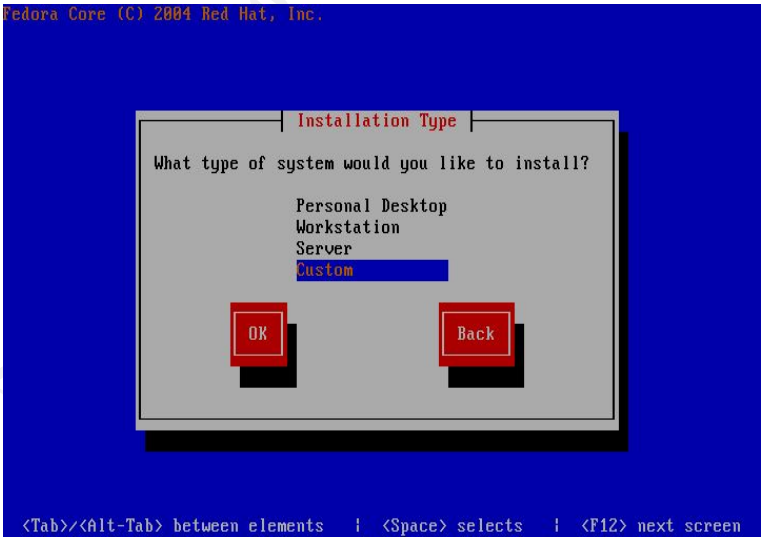

Figure 7 – Custom Installation Type

By choosing this type of installation, users will become able to install only desired packages.

6. Disk Partitioning, the table below explains how this is going to be configured for repository server machine, the partitions were created in same order as showed:

Mount Point	*Size(MB)*	*Explanation*
Swap	2 x 512(RAM)	Swap partition equals to twice the amount of RAM you have on the system
/boot	100	This partition contains the operating system kernel, along with files used during the bootstrap process
/	5000	This is the “root” partition
/var	All left space	All repository files will reside on this partition

These values are based on recommended partitioning scheme[10]. With 512MB of RAM, Swap partition will size 1024MB, /boot partition creation is necessary for easily maintain and configure possible multiple Linux installations and for old PC BIOS compatibility.

On /var partition, it will be stored all large files including RPM packages, which will be part of repository server files, this partition will size almost 74GB since the disk has 80GB of total space.

7. Next screen is about Boot Loader Configuration, choose “Grub” as loader and pick up a password for avoiding unauthorized changes on configuration at boot time. Figure 8 shows that screen:

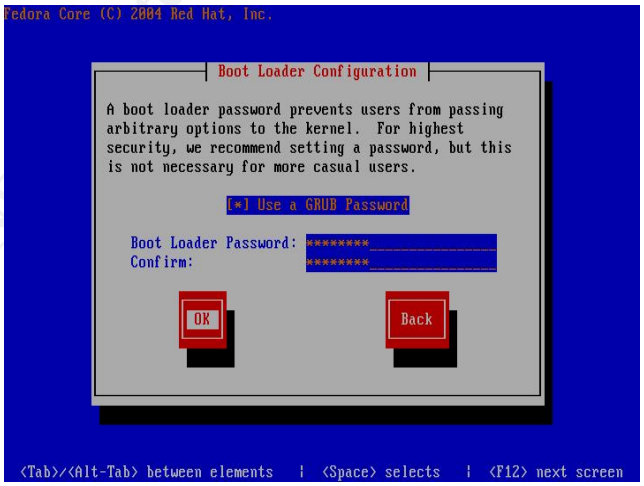

Figure 8 – Grub Password

10 Red Hat Linux x86 Installation Guide, Recommended Partitioning Scheme. URL: http://redhat.com/docs/manuals/linux/RHL-9-Manual/install-guide/s1-diskpartitioning.html(14 Sep. 2004).

8. Network Configuration. Here, Iuri has chosen IP address 192.168.1.200 and hostname “proteus” for the server. The gateway IP address is 192.168.1.12 and name servers addresses are 192.168.1.24 and 192.168.1.29, Figure 9 and 10 show these installation screens:

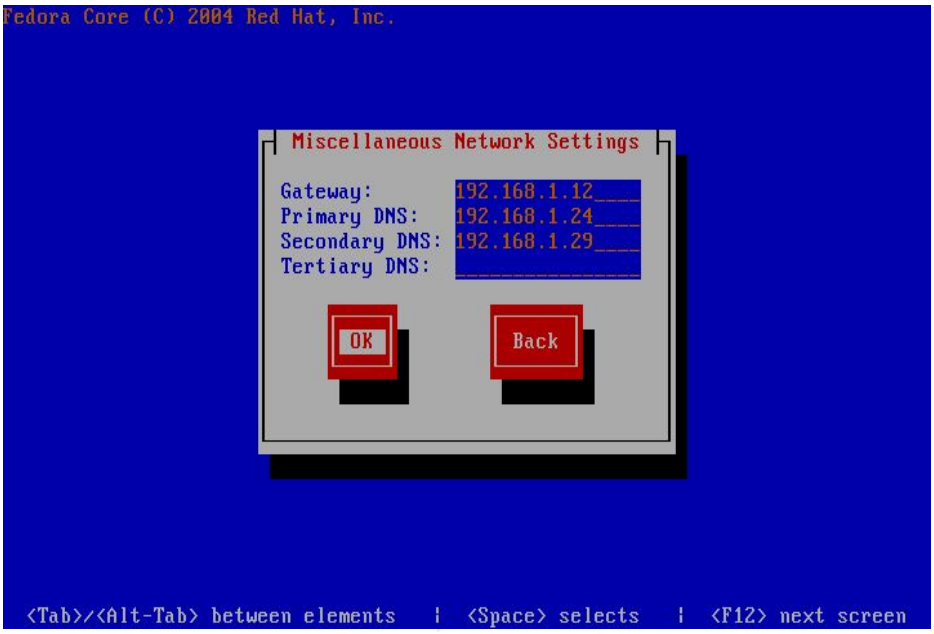

Figure 9 – Server's IP Address

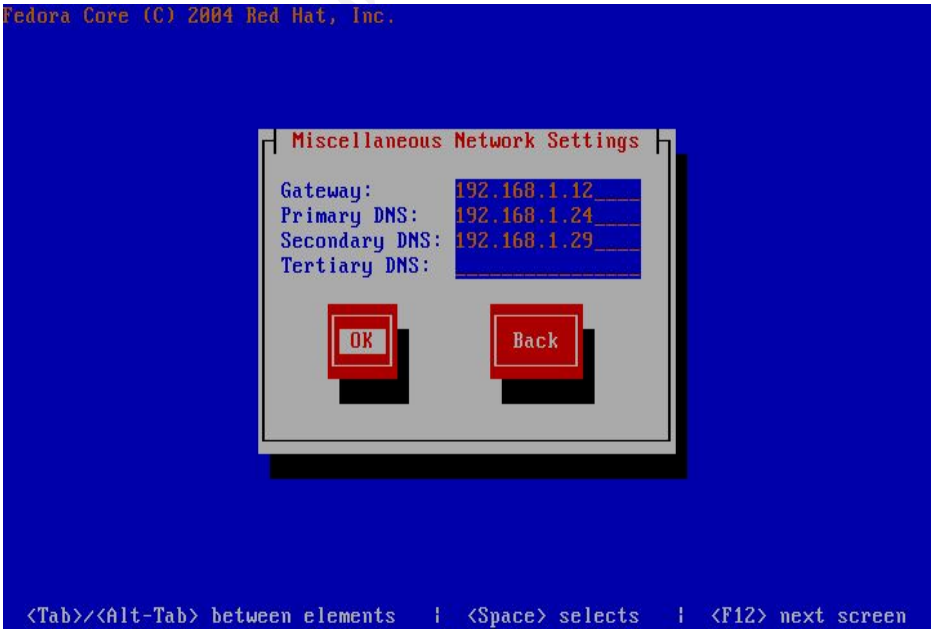

Figure 10 – Gateway and DNS Configurations

9. Next step is about Firewall activation, choose “Enable Firewall” option and then OK. No rules will be custom made at this point, default configuration is enough since this blocks all incoming connections except:

   - Loopback traffic(lo device), this is necessary for internal communication between some operating system processes

   - 50, 51 protocols(Crypt and Auth headers, respectively), necessary for some types of VPN(Virtual Private Networks) implementation

   - ICMP protocol, necessary for network testing such as *ping* command

10. Next configuration screens ask information about additional languages support and Timezone configuration, for this, Iuri has chosen “UTC – America/Sao Paulo”, this is the location where GIAC Enterprises is based, the next section is about *root* user password, choose a password and continue.

11. The last stage of Installation process is about Package Selection, leave all check boxes empty for installing only minimal packages as showed on Figure 11, this action will utilize only about 562MB of disk space. Other packages will be installed separately during implementation.

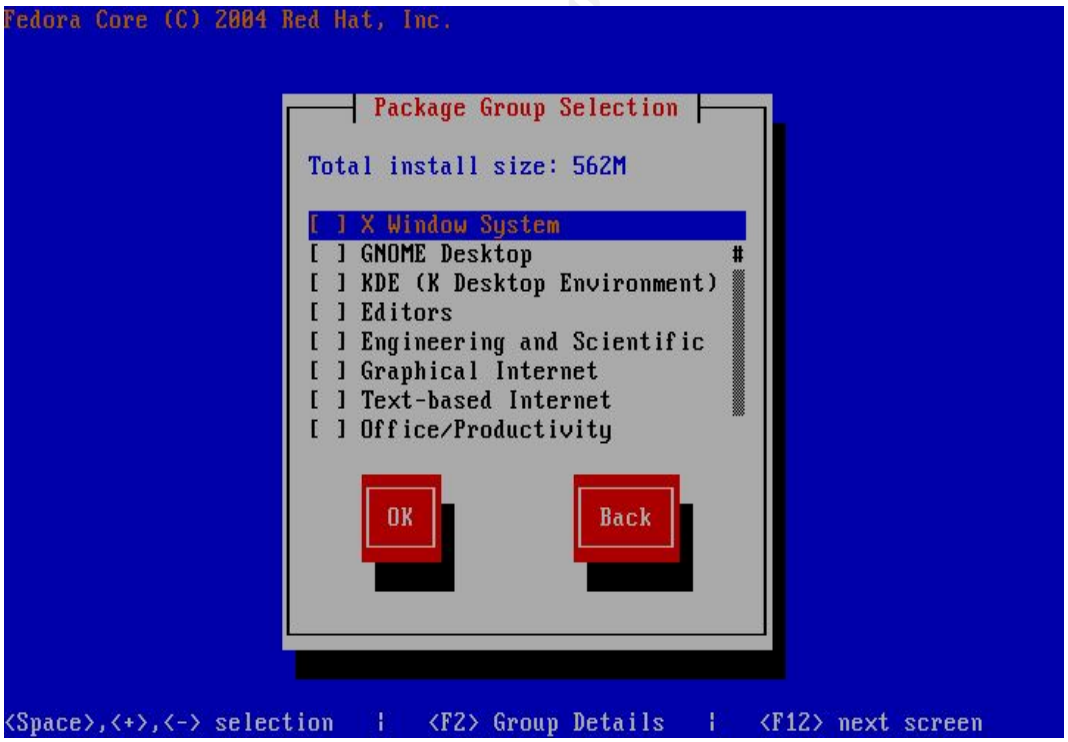

Figure 11 – Package Selection

12. To finish Installation process, remove the media from the drive and choose “Reboot” option.

Post Installation Steps

At this stage, a minimal installation of Fedora Core 2 Linux distribution is ready to run, but before beginning the implementation of repository software, there are some post-install steps that Iuri has to do. Note that all steps executed from this point are executed as *root* user if not clearly stated.

1) Installation Logs Backup: Files *anaconda-ks.cfg*, *install.log* and *install.log.syslog* in /root directory must be copied for some safety place outside server machine.

   The first one is a configuration file used by *kickstart* tool, this tool enables administrators to apply the same install configurations and settings on another installation session by simply passing this file as an argument to *linux*(kernel image) first installation command.

   It's a good practice storing copies of those files for possible future analysis, the copies can be made by using *ftp*, *wget* or *scp* programs.

2) Useless Package Removal: Even when leaving all packages groups check boxes empty on installation process, some packages that won't be useful for this solution will be installed by Fedora installer. So, it's a good practice removing them in order to minimize the chances of a software bug exploration.

   First remove all packages that can lead to remote exploitation and tools that are commonly used for attackers that are not going to be used on implementation: network services, terminal emulators, network tools such as *telnet*, *ftp*, *nc* and *nmap*.

   By issuing the commands below, Iuri is able to identify which network services are enabled by default, if the service is not going to be used, it is then removed. There's no need to hardening services at this point, since the firewall is enabled.

```
[root@proteus root]# netstat -anp | grep "LISTEN \|^udp"
tcp   0   0   0.0.0.0:32768        0.0.0.0:*   LISTEN   1557/rpc.statd
tcp   0   0   0.0.0.0:111          0.0.0.0:*   LISTEN   1538/portmap
tcp   0   0   127.0.0.1:25         0.0.0.0:*   LISTEN   1734/sendmail:
tcp   0   0   :::22   :::*                     LISTEN   1705/sshd
udp   0   0   0.0.0.0:32768        0.0.0.0:*             1557/rpc.statd
```

   Based on output above, there are four network services currently running on the server: *statd*, *portmap*, *sendmail* and *ssh* daemon. The last one is the only network service that will be used on this implementation, so, other network services will be removed with "rpm -e package" command.

   Another good place to look for useless software is on */etc/init.d* directory, this directory stores many network services startup scripts, to know of which package each script belongs, executes the following rpm command:

```
[root@proteus root]# rpm -qf /etc/init.d/*
```

   The output of the last command will be something like the one showed below:

```
acpid-1.0.2-6
anacron-2.3-30
apmd-3.0.2-22
at-3.1.8-53
initscripts-7.53-1
gpm-1.20.1-49
iptables-1.2.9-2.3.1
...
```

Based on *netstat* and *rpm* query command output, Iuri is able to remove some RPM packages that will not be used on this implementation:

```
[root@proteus root]# rpm -e autofs isdn4k-utils nscd pcmcia-cs nss_ldap
[root@proteus root]# rpm -qf `which rpc.statd`
nfs-utils-1.0.6-20
[root@proteus root]# rpm -e nfs-utils
[root@proteus root]# rpm -e portmap
error: Failed dependencies:
        portmap is needed by (installed) ypbind-1.17.2-1
[root@proteus root]# rpm -e portmap ypbind
error: Failed dependencies:
        ypbind is needed by (installed) yp-tools-2.8-3
[root@proteus root]# rpm -e portmap ypbind yp-tools
[root@proteus root]# rpm -e sendmail
error: Failed dependencies:
        smtpdaemon is needed by (installed) mdadm-1.5.0-3
[root@proteus root]# rpm -e sendmail mdadm
```

Some packages have dependencies, when these dependencies are useless, they are also removed. The last place to look for more useless packages are in local RPM database.

The command below queries all installed packages and redirects the output for file *rpm-a.txt*. After this, the file is reviewed and then removal action is taken:

```
[root@proteus root]# rpm -qa > rpm-a.txt
[root@proteus root]# rpm -e ash dhclient dos2unix finger libpcap ppp \
wvdial rp-pppoe make stunnel mtools syslinux mkbootdisk nc dosfstools \
pam_smb quota ed rdist rsh talk rsync minicom tcsh telnet ftp \
traceroute jwhois wireless-tools zip stunnel
```

NOTE: Additional RPM packages essential for repository implementation will be installed on demand, this makes easy when looking for all information involving a specific topic such as SSH or Apache.

3) Users and Groups Creation: There will be created 2 users, one normal and unprivileged user for executing custom repository script and one administrator user, which will be able to execute general administration tasks by using *sudo*[11] command such as “sudo sh” for getting a superuser shell.

11 Sudo allows certain users the ability to run some (or all) commands as root or another user. URL: http://www.courtesan.com/sudo/ (18 Sep. 2004).

Users names will be chosen based on random data for reducing the chances of successfully brute force attacks or information discovery attempts(user tasks, position, etc) based on user's login name.

The first user will be member of *repoadm* group or Repository Administrators Group, which will be created, the following commands will add the user/group and setup user's password:

```
[root@proteus root]# groupadd repoadm
[root@proteus root]# grep ^repoadm: /etc/group
repoadm:x:500:
[root@proteus root]# useradd -g repoadm -n u300
[root@proteus root]# grep ^u300: /etc/passwd
u300:x:500:500::/home/u300:/bin/bash
[root@proteus root]# ls -l /home/
drwx------  2 u300 repoadm 4096 Sep 10 21:14 u300
[root@proteus root]# passwd u300
Changing password for user u300.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

Note that the “-n” command parameter states that no group with the same name of the user will be created, in comparison to Fedora's standard behavior.

On modern Linux systems, users id automatically start from 500, values between 0 and 99 are typically reserved for system accounts such as *bin*, *daemon* and *news* (see *useradd* man page for more details).

For the administrator user, the same creation process is applied, the user will also be member of repository administrators group and *wheel* group, which is created by default and used as an administrative group. The primary group will have the same name of the user:

```
[root@proteus root]# useradd -G repoadm u401
[root@proteus root]# groups u401
u401 : u401 wheel    repoadm
[root@proteus root]# passwd u401
```

The group called “wheel”, which is created by default on Linux, will have some administrative privileges such as */bin/su* command execution, which will be explained later.

SUDO Installation and Configuration

SUDO stands for “superuser do”, it will be installed so that Iuri will become able to restrict superuser's actions for only users who are defined to do it. The package is located on first distribution CD, below are the commands executed in order to install it:

```
[root@proteus root]# mount /mnt/cdrom/
```

```
[root@proteus root]# cd /mnt/cdrom
[root@proteus cdrom]# rpm -ivh Fedora/RPMS/sudo-1.6.7p5-26.i386.rpm
warning: Fedora/RPMS/sudo-1.6.7p5-26.i386.rpm: V3 DSA signature: NOKEY,
key ID 4f2a6fd2
Preparing...         ########################################### [100%]
```

The warning message is about GPG verification, it's done by default when invoking rpm install command. In order to validate a package, the public key of package creator must be installed. To install Fedora's Public Key and verify its installation, issue the commands:

```
[root@proteus root]# rpm --import /usr/share/rhn/RPM-GPG-KEY-fedora
[root@proteus root]# rpm -q gpg-pubkey
```

In order to give administrator privileges for user *u401*, Iuri must specify this in */etc/sudoers* file, this the main configuration file of SUDO. The configuration is straightforward, the basic syntax follow the statement: “Who can execute What command(s) and Where”.

On repository server, only *u401* user must be able to run commands as *root* or superuser, to make this possible, execute the command ***visudo*** for editing *sudoers* file. The configuration file must have the line below:

```
u401    proteus=(ALL)   ALL
```

To validate the configuration, run any command following the format “sudo command” as showed below, execution of *iptables*[12] command for listing active firewall rules:

```
[u401@proteus u401]$ /sbin/iptables -L OUTPUT -n
iptables v1.2.9: can't initialize iptables table `filter': Permission
denied (you must be root)
[u401@proteus u401]$ sudo /sbin/iptables -L OUTPUT -n
Password:
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[u401@proteus u401]$ sudo /sbin/iptables -L FORWARD -n
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0           0.0.0.0/0
```

As showed on example, when user tries to execute *iptables* command, the system complains about insufficient privileges. This happens because only *root* is able to access necessary system resources for *iptables* command execution. Also note that when *sudo* command is called for the first time by a user, the system asks for user's password(not root's one) and then the system validates a “secure session” for a period of time(5 minutes unless overridden in *sudoers* file).

Another advantage of using SUDO is auditing support. All SUDO actions

12 Netfilter and iptables are building blocks of a framework inside the Linux 2.4.x and 2.6.x kernel. URL: http://www.netfilter.org/ (18 Sep. 2004).

are logged, such as failed commands and password attempts, this feature is in accordance with checklist item "Logging Enable Implementation". Additionally, it will be made a little, but important modification on default logging behavior of SUDO, which have its logs appended to */var/log/secure* file. This file is usually used by another applications such as *pam* and *useradd*, it's a good practice having separate logs, administrator can apply different permissions and configurations for each log file. By adding the line below on *sudoers* file, it will make SUDO logs been added to */var/log/sudolog* file:

```
Defaults        syslog=auth, logfile=/var/log/sudolog
```

The output below illustrates an example of a SUDO log entry where the user *u300* failed on authentication. Log information includes, date, time, user, event and the command executed:

```
[root@proteus log]# cat /var/log/sudolog
Sep 11 21:16:51 : u300 : 3 incorrect password attempts ; TTY=pts/3 ;
PWD=/home/u300 ; USER=root ; COMMAND=/sbin/iptables -L -n
```

In order to successfully start logging to a different location, issue the commands below for creating and configuring right permissions on logfile:

```
[root@proteus log]# touch /var/log/sudolog
[root@proteus log]# chmod 600 /var/log/sudolog
```

After installing and configuring SUDO, Iuri has configured *logrotate* program for rotating SUDO log file, *logrotate* is easy and simple to configure, configurations are located in */etc/logrotate.conf* file and */etc/logrotate.d* directory. To configure SUDO log rotating, add the lines below to "*/etc/logrotate.conf*" file:

```
/var/log/sudolog {
        create 0600 root root
        compress
        nomail
        missingok
        notifempty
        rotate 5
        size 5M
}
```

This will rotate */var/log/sudolog* file every time it sizes 5MB and keeping 5 compressed log copies of this file with only reading and writing permission for *root* user.

NOTE: SUDO installation is intended to be a hardening procedure, this is done before main repository implementation steps because the majority of these steps will need *root* access and commands execution. See the advantages below.

Security Advantages of SUDO Implementation

SUDO implementation is valuable since Iuri will be able to control *root* user

access and actions and other simple methods of superuser access can be disabled such as *telnet, ssh, rlogin* commands and console login. Auditing support is another important feature, as showed above, all interesting actions will be logged to a specific file.

The Hardening process will continue along this document. Source code of all custom scripts and configuration files such as *sudoers* file are described on Appendix B.

Disabling Root Access

There are many standard methods of getting superuser access on Linux, some of these are not safety since passwords can be discovered. After installing SUDO, which is a safety method, Iuri will disable directly shell access by changing superuser's shell to "/sbin/nologin". This action will make *root* user unable to get shell access by using standard methods such as console login, ssh and telnet.

Configuring */bin/su* Access

Iuri's idea is to allow only *u401* user who is member of *wheel* group accessing superuser resources. But he has found that a normal user could "su to root", even with superuser's shell configured to "/sbin/nologin" as showed below:

```
[u300@proteus u300]$ su – root
Password:
This account is currently not available.
[u300@proteus u300]$ su -s /bin/bash – root
Password:
[root@proteus root]#
```

That's why *u401* user is member of "wheel" administrative group. Iuri has configured PAM[13] settings for *su* command, so only users on *wheel* administrative group will be able to execute *su* command. The output below correspond to "/etc/pam.d/su" file content, which holds settings about *su* command:

```
#%PAM-1.0
auth     sufficient /lib/security/$ISA/pam_rootok.so
# Uncomment to implicitly trust users in the "wheel" group.
auth     sufficient /lib/security/$ISA/pam_wheel.so trust use_uid
# Uncomment to require a user to be in the "wheel" group.
#auth    required   /lib/security/$ISA/pam_wheel.so use_uid
auth     required   /lib/security/$ISA/pam_stack.so service=system-auth
account  required   /lib/security/$ISA/pam_stack.so service=system-auth
password required   /lib/security/$ISA/pam_stack.so service=system-auth
session  required   /lib/security/$ISA/pam_stack.so service=system-auth
session  optional   /lib/security/$ISA/pam_selinux.so multiple
session  optional   /lib/security/$ISA/pam_xauth.so
```

Using the configuration file above users that are not on *wheel* group will not be able to successfully execute *su* command.

13 Pluggable Authentication Modules, URL: http://kernel.org/pub/linux/libs/pam/ (22 Sep. 2004).

Repository Server Implementation

There are many ways of packages downloading and synchronizing such as *rsync* method, but the problem is that not everyone or every machine has *rsync* connection open to outside world, even FTP access is blocked on some networks.

WWW port(80) is usually open, users need Internet access. Sometimes this traffic is controlled by a proxy server, and maybe only a little list of sites are available for browsing, thus Iblio's hostname, which is the Internet package repository chose by Iuri, must be authorized for everything goes fine. Proxy support is also available on *yum_repository* script, also with authentication support.

The standard connection using 80 port is one good feature, another script feature is automated security checks(GPG keys and MD5) and repository tree building and updating. Every downloaded packages that fails on “rpm -K” command is moved to “.BAD” extension and becomes excluded from repository tree building process, since only “.rpm” extension files are read by *yum-arch* and *createrepo* programs, thus bad packages don't become available for users.

Some safety checks are also implemented such as multiple instances running on the same time, writing permissions to repository directories and software requirements checking. *Wget* program will only download new packages, if a file is not completely downloaded, it will try to resume the action from already downloaded part(“-c” parameter) if exists, consuming less bandwidth.

Custom Script Installation and Configuration

Next step is about *yum_repository.sh* script installation, this script will be responsible for reading some user defined variables and then executes its main tasks: keep repository packages synchronized with a server mirror on Internet and update repository tree.

It's a simple shell script that will be running as a *cron job*, it calls *wget* program for packages download and runs *yum-arch* or *createrepo* programs, for repository structure creation. Since Fedora Core 3 release, *yum* program needs *createrepo* repository database type, in older versions of Fedora, yum client needs *yum-arch* repository database type. For this document, Fedora Core 2 is the target distribution version, so *yum-arch* repository type will be used. Note that the script is compatible with both methods of repository structure, it will work on networks with any version of Fedora distribution.

For script configuration, some questions have to be discussed before:

- Repository server will store only update packages?
- What versions of Linux it will support?
- From which mirror server RPM packages will be downloaded?
- Downloaded packages will be validated and verified on the server?

Iuri has decided to store both update packages and installation medias RPM packages. This is useful since users will be able to install new packages by using *yum* interface and without needing the CD media locally mounted. Iuri will also create a "plus" or extra repository, which will provide RPM packages that are not found on Fedora Linux distribution such as development tools, browser plugins and other interesting software.

On GIAC Enterprises all Linux clients are based on the second version of Fedora Linux distribution, so the repository server will only support Fedora Core 2 version. All released update packages for this distribution will be made available on repository server.

Fedora packages are mirrored on many server around the world, Iuri has chosen to use the server at Iblio's web server:

http://distro.ibiblio.org/pub/linux/distributions/fedora/linux/core/updates/2/

The above URL points to Fedora Core 2 update packages, these packages will be stored on repository server becoming available to Internal network users. Iuri will create 3 types of repository: "os", "updates" and "plus".

The first one is a reference to RPM packages that come with Fedora distribution medias, so it's content is static, never changes since Iuri will copy all packages for this repository directory only once.

The repository named "updates" will store only update RPM packages, these packages are released by Fedora Project community as new bugs and software improvements are discovered and implemented. From a security standpoint, that's the most important data that has to be available for patch management solution, without this information users can not update their Linux systems. This repository is automatically populated by *yum_repository* script and Iuri will enable GPG and MD5sum automatically checks on this repository.

The last repository will store extra packages, packages that are not shipped within Fedora distribution but are very important for GIAC Enterprises Linux users. The "plus" repository content is manually populated by Iuri.

The script is available on Internet, before downloading the script Iuri has configured environment variable "http_proxy" for pointing to proxy server of testing network and then used *wget* program for downloading the script:

```
[u401@proteus u401]$ export http_proxy=192.168.1.12:3128
[u401@proteus u401]$ wget
http://fedoranews.org/alex/scripts/yum_repository.sh
00:07:36 (27.19 KB/s) - `yum_repository.sh' saved [8,879/8,879]
```

NOTE: Starting from this point, all commands will be executed using sudo program executed by *u401* user when superuser access is necessary, those commands will be logged to *sudolog* file in */var/log* directory.

This script will be located on */usr/local/bin* directory, this is the default

location for customized executable programs on Iuri's Linux system. That directory is only writable by *root* user, so *sudo* program is used to copy the script for its final location:

```
[u401@proteus u401]$ sudo mv yum_repository.sh /usr/local/bin/
Password:
[u401@proteus u401]$ cd /usr/local/bin/
[u401@proteus bin]$ ls -l yum_repository.sh
-rw-rw-r--  1 u401 u401 8879 Sep 20 23:49 yum_repository.sh
```

The script permissions will be configured for only allowing *root* for editing the file and only Repository Admins group for executing it. The file owner will be *root* and group will be *repoadm*:

```
[u401@proteus bin]$ sudo chown root:repoadm yum_repository.sh
[u401@proteus bin]$ sudo chmod 650 yum_repository.sh
[u401@proteus bin]$ ls -l yum_repository.sh
-rw-r-x---  1 root repoadm 8879 Sep 20 23:49 yum_repository.sh
```

Now, the script is ready to be edited, based on answers already described above, Iuri opens the file with *vi* editor and edit some variables inside the script:

```
[u401@proteus bin]$ sudo vi yum_repository.sh
```

Just one automatically repository will be configured, this will point to Fedora Core 2 update packages, the files will be downloaded from Iblio's server and will be located in directory */var/repository/linux/fedora/2/updates* on repository server's local filesystem, so *MIRROR_URL* and *MIRROR_DIR* shell array variables will be configured as showed below:

```
MIRROR_URL[0]="http://distro.ibiblio.org/pub/linux/distributions/fedora/linux/core/updates/2/"
MIRROR_DIR[0]="/var/repository/linux/fedora/2/updates/"
```

NOTE: The script will only run against mirrors web servers that implement Apache's “Indexes” option or equivalent. This enable users to browse directories without an index page, thus listing directories content.

The script comes with two repository entries enabled by default, but only one showed above has to be enabled, other entries must be disabled by issuing a “#” symbol on beginning of the line.

The next setting is about the proxy configuration, two variables must be edited: *PROXY_SERVER* and *PROXY_PORT.*

```
PROXY_SERVER="192.168.1.12"
PROXY_PORT="3128"
```

The IP address of proxy server on testing network is 192.168.1.12 and proxy server's port is 3128. The script has support for proxy authentication, but this is not applicable for this solution, so *PROXY_USER* and *PROXY_PASS* variables are left blank.

There's a variable called *DEF_UMASK* within script configuration, the value of this variable represents default files *umask* when creating repository packages on */var/repository* directory. Iuri wants to enable all users that are members of *repoadm* group to manage repository files, so this *umask* value will be configured as showed below:

```
DEF_UMASK=003
```

By using the umask value of 003, every created file on repository will have read and write permissions for both owner and group, and only read permission for "other", that is the necessary permission values for repository server.

The variable GPGCHECK has value of 1 by default, it means that automatically GPG and MD5 checks will be executed, this action is taken before repository's tree creation process and just after packages download. If a file is corrupted or GPG signature check fails, the file is renamed with ".BAD" extension and the file is excluded from repository files available for users.

After saving the script with the correct values showed above, there are simple actions to be taken, these are about correct file permissions and script logs maintenance.

The repository script will run as a *cron job*, so it will generate too much log information, Iuri will enable *logrotate* program for rotating script's log, which will be located in */var/log/yum_repository.log* file, the configuration below will be added to file */etc/logrotate.conf*:

```
/var/log/yum_repository.log {
        create 0664 root repoadm
        compress
        nomail
        missingok
        notifempty
        rotate 2
        size 5M
}
```

This will rotate */var/log/yum_repository.log* file every time it sizes 5MB and keeping 2 compressed log copies of this file with read and write permissions for *root* user and *repoadm* group, and only read permission for "other". The log file does not exist, so it will be created and right permissions will be applied by issuing the following commands:

```
[u401@proteus bin]$ sudo touch /var/log/yum_repository.log
[u401@proteus bin]$ sudo chmod 664 /var/log/yum_repository.log
[u401@proteus bin]$ sudo chown root:repoadm /var/log/yum_repository.log
```

Repository tree root directory will be */var/repository* and Fedora Core 2 update packages will reside on */var/repository/linux/fedora/2/updates* directory, the same upper directory structure applies to "plus" and "os" repositories. The following

output illustrates the complete creation and configuration of those directories:

```
[u401@proteus /]$ sudo mkdir -p /var/repository/linux/fedora/2/updates
[u401@proteus /]$ sudo mkdir /var/repository/linux/fedora/2/os
[u401@proteus /]$ sudo mkdir /var/repository/linux/fedora/2/plus
[u401@proteus /]$ sudo chown -R root:repoadm /var/repository/
[u401@proteus /]$ sudo chmod -R 775 /var/repository/
[u401@proteus /]$ sudo chmod -R g+s /var/repository/
```

SGID bit controls the "set group id" status of a file or directory. This behaves the same way as SUID, except the group is affected instead. If you set the SGID bit on a directory (“chmod g+s directory"), files created in that directory will have their group set to the directory's group[14]. By doing this action Iuri will guarantee *repoadm* group privileges on repository files.

Configuring “updates” repository as a *cron job*

After script installation and post-install steps taken, it's time to executes it. To test script functionality runs it on foreground using *u300* user, who is member of *repoadm* group:

```
[u300@proteus u300]$ /usr/local/bin/yum_repository.sh
```

This command may take several hours to finish, depending on Internet link speed. If no error message appears, the script must be configured to run as a *cron job*, by every hour. This is done by editing a file in */etc/cron.d* directory as follows:

```
[u401@proteus u401]$ sudo vi /etc/cron.d/yum_repository.cron
```

The content of this file will make *cron* program disabling mail feature and running the script command as *u300* user every hour on its first minute. The content of *yum_repository.cron* file is showed below:

```
MAILTO=""
01 * * * * u300 /usr/local/bin/yum_repository.sh
```

Creating “os” repository: RPM base packages

The “os” repository will store base RPM packages, those shipped with Fedora distribution ISO images, by providing base packages, users will be able to install any software that comes with Fedora by simple executing “yum install program” without needing to mount CD ROM medias. If the target package has dependencies, *yum* will look for them automatically, it will provide interactive “yes/no” questions for confirmation of install, remove and update commands.

To create “os” repository, Iuri must copy all RPM packages founded on Installation CD ROM medias and then execute *yum-arch* command for creating *headers* directory, which is part of repository structure. All base RPM packages will

14 Users, Groups and User-Private Groups.
URL: http://redhat.com/docs/manuals/linux/RHL-5.1-Manual/manual/doc077.html (24 Sep. 2004)

need about 2GB of hard disk space, they are located in “Fedora/RPMS/” path within four Installation CD ROM medias.

The following commands must be executed after all RPM packages were copied to */var/repository/linux/fedora/2/os* directory, note that the first one need superuser privileges, so it's executed using SUDO:

```
[u401@proteus /]$ sudo chmod -R 775 /var/repository/linux/fedora/2/os/
[u401@proteus /]$ sudo chmod g+s /var/repository/linux/fedora/2/os/
[u401@proteus /]$ cd /var/repository/linux/fedora/2/os/
[u401@proteus os]$ sudo -u u300 yum-arch .
Digesting rpms 100 % complete: zsh-html-4.2.0-1.i386.rpm

   Total: 1619
   Used: 1619
   Src: 0

Writing header.info file
```

The “os” repository data does not change, since it's based on base rpm packages, so it doesn't need to be updated, those actions are done only once.

Creating “plus” repository: Extra RPM packages

Iuri knows that users will need to install non standard RPM packages, packages that are not built in Fedora distribution tree. The “plus” repository will be made for storing these kind of packages. There are many RPM repositories around the net, where good and useful RPM packages can be found, one of the famous packages search portal is “rpm.pbone.net” Website[15].

The following steps will illustrate how to make *Firefox*[16] web browser RPM available through “plus” repository:

Manually checking the extra package with GPG key

In order to validate the extra package, Iuri needs to know who is the package creator and then installs his public GPG key. If the verification(“rpm -K” command) runs OK then *yum-arch* program is called for creating headers data structure. The following actions must be done for every package added to “plus” repository tree.

NOTE: If package creator provides a GPG public key, this absolutely does not guarantee that the package content is safety or does not contain dangerous code. This only validate the source of the package.

The *Firefox* browser package was copied from the URL below:

http://rpm.pbone.net/index.php3/stat/4/idpl/1239065/com/firefox-0.8-3.1.fc2.dag.i386.rpm.html

On above page, there're many information about the package including a

15 http://rpm.pbone.net/index.php3/stat/5 (27 Sep. 2004)
16 http://www.mozilla.org/products/firefox/ (28 Sep. 2004)

link to vendor or package creator, in this case, Dag Wieers[17]. On his website is easy to find the public GPG key used to sign packages on URL below:

http://dag.wieers.com/packages/RPM-GPG-KEY.dag.txt

To download and install the key, executes the following commands:

```
[u401@proteus /]$ cd /tmp/
[u401@proteus tmp]$ export http_proxy=192.168.1.12:3128
[u401@proteus tmp]$ wget http://dag.wieers.com/packages/RPM-GPG-
KEY.dag.txt
21:36:19 (44.03 KB/s) - `RPM-GPG-KEY.dag.txt' saved [1,672/1,672]
[u401@proteus tmp]$ sudo rpm --import RPM-GPG-KEY.dag.txt
[u401@proteus tmp]$ rpm -q gpg-pubkey
gpg-pubkey-4f2a6fd2-3f9d9d3b
gpg-pubkey-6b8d79e6-3f49313d
[u401@proteus tmp]$ rpm -qi gpg-pubkey-6b8d79e6-3f49313d
Name        : gpg-pubkey                    Relocations: (not
relocatable)
Version     : 6b8d79e6                          Vendor: (none)
Release     : 3f49313d                      Build Date: Sun 21 Nov 2004
09:36:34 PM BRST
Install Date: Sun 21 Nov 2004 09:36:34 PM BRST      Build Host:
localhost
Group       : Public Keys                   Source RPM: (none)
Size        : 0                                License: pubkey
Signature   : (none)
Summary     : gpg(Dag Wieers (Dag Apt Repository v1.0)
<dag@wieers.com>)
```

The package will be downloaded from a FTP site, the following commands will make this task:

```
[u401@proteus tmp]$ cd /var/repository/linux/fedora/2/plus/
[u401@proteus plus]$ export ftp_proxy=192.168.1.12:3128
[u401@proteus plus]$ wget
ftp://ftp.freshrpms.net/pub/dag/fedora/2/en/i386/RPMS.dag/firefox-0.8-
3.1.fc2.dag.i386.rpm
```

For manually checking *Firefox* RPM execute the commands:

```
[u401@proteus plus]$ rpm -K firefox-0.8-3.1.fc2.dag.i386.rpm
firefox-0.8-3.1.fc2.dag.i386.rpm: (sha1) dsa sha1 md5 gpg OK
```

The above output shows that the package is OK, so it's ready to be included in repository structure and becoming available to users:

```
[u401@proteus plus]$ sudo chown root:repoadm firefox-0.8-
3.1.fc2.dag.i386.rpm
[u401@proteus plus]$ sudo chmod 775 firefox-0.8-3.1.fc2.dag.i386.rpm
[u401@proteus plus]$ sudo -u u300 yum-arch .
```

17 http://dag.wieers.com/ (28 Sep. 2004)

```
Digesting rpms 100 % complete: firefox-0.8-3.1.fc2.dag.i386.rpm
   Total: 1
   Used: 1
   Src: 0
Writing header.info file
```

Server Side Configuration and Updating using YUM

All available RPM packages were downloaded and *yum_repository.sh* script keeps downloading new packages as soon as they become available on Internet Fedora's mirror. On client side, yum settings can be configured to search for RPM packages in 3 different locations: local disk, a web server or a ftp server. Thus, the server will look for packages on his own file system, this is done by issuing an URL into */etc/yum.conf* file.

At this point, only the repository server is able to configure yum client, the web server package will be installed with “yum install” command demonstrating the great advantage of using yum interface for RPM management. After web server preparation, users will become able to update their systems using network connection.

The configuration file for yum clients is very simple, the *[main]* entry is used to define global settings such as log file location and debug level, this values will be extended to all other entries that don't have their own setting defined. Other entries named different than “main” will be parsed as repository entries. Default configuration comes with *[base]* and *[updates]* entries.

Since the repository server stores all base and update packages on his own disk, the default location pointing to Fedora Project's server can be removed. The following configuration will set yum for fetching package data on local disk:

```
# yum.conf on repository server
[main]
cachedir=/var/cache/yum
debuglevel=2
logfile=/var/log/yum.log
pkgpolicy=newest
distroverpkg=redhat-release
tolerant=1
exactarch=1
retries=20

[base]
name=Fedora Core $releasever - $basearch – Base
baseurl=file:///var/repository/linux/fedora/$releasever/os/

[updates-released]
name=Fedora Core $releasever - $basearch - Released Updates
```

```
baseurl=file:///var/repository/linux/fedora/$releasever/updates/

[plus]
name=Fedora Core $releasever - $basearch – Extra Packages
baseurl=file:///var/repository/linux/fedora/$releasever/plus/
```

The parameter “name” is a simple description of the repository entry and “baseurl” parameter refers to where the package data will be found. The yum variable *$releasever* is very important, its value stores Fedora release version, in this case its value will be 2. It's a good practice using the variable instead of a static integer, thus Iuri will not have to change yum settings on all machines when they upgrade their Fedora release version(dist-upgrade).

With that configuration active, Iuri is ready to update repository server by issuing the command below:

```
[u401@proteus u401]$ sudo yum update
Gathering header information file(s) from server(s)
Server: Fedora Core 2 - i386 – Base
Server: Fedora Core 2 - i386 - Released Updates
Finding updated packages
Downloading needed headers
... headers downloading ...
Resolving dependencies
Dependencies resolved
I will do the following:
[install: kernel 2.6.9-1.3_FC2.i686]
[update: kudzu 1.1.68.2-1.i386]
[update: tzdata 2004e-1.fc2.noarch]
[update: slang 1.4.9-12.i386]
[update: net-tools 1.60-25.1.i386]
[update: glibc 2.3.3-27.1.i686]
[update: glibc-common 2.3.3-27.1.i386]
[update: tcpdump 14:3.8.2-6.FC2.1.i386]
[update: initscripts 7.55.1-1.i386]
[update: libuser 0.52.5-0.FC2.1.i386]
[update: hwdata 0.120-1.noarch]
[update: libxml2 2.6.16-2.i386]
[update: lha 1.14i-14.1.i386]
[update: glib2 2.4.7-1.1.i386]
[update: cyrus-sasl 2.1.18-2.2.i386]
[update: cyrus-sasl-md5 2.1.18-2.2.i386]
[update: libxml2-python 2.6.16-2.i386]
[update: cyrus-sasl-plain 2.1.18-2.2.i386]
[update: info 4.7-4.i386]
[update: man 1.5o1-6.i386]
[update: system-config-network-tui 1.3.17-0.FC2.1.noarch]
[update: krb5-libs 1.3.4-6.i386]
Is this ok [y/N]: y
```

Iuri has answered “y” to above question making yum updating repository's operating system. Note that the newest kernel package is installed and old version are kept on the system, this is the default behavior and can be configured on yum configuration file.

For loading the new installed kernel, Iuri reboots repository machine and loads the new kernel image on GRUB boot screen, since booting process has terminated with no errors the old kernel package can be removed, when this action is taken, yum reconfigures GRUB configuration so that all settings match the current state of the system.

As showed on output below, now repository machine is updated and the latest kernel is loaded:

```
[u401@proteus u401]$ sudo yum update
Password:
Gathering header information file(s) from server(s)
Server: Fedora Core 2 - i386 – Base
Server: Fedora Core 2 - i386 - Released Updates
Finding updated packages
Downloading needed headers
No Packages Available for Update
No actions to take
[u401@proteus u401]$ uname -a
Linux proteus 2.6.9-1.3_FC2 #1 Mon Nov 10 14:46:43 EST 2004 i686 athlon
i386 GNU/Linux
[u401@proteus u401]$ sudo rpm -q kernel
kernel-2.6.5-1.358
kernel-2.6.9-1.3_FC2
[u401@proteus u401]$ sudo rpm -e kernel-2.6.5-1.358
```

Web Server Installation and Hardening

To install Apache web server, Iuri simple issue the command below:

```
[u401@proteus u401]$ sudo yum install httpd
Password:
Gathering header information file(s) from server(s)
Server: Fedora Core 2 - i386 – Base
Server: Fedora Core 2 - i386 - Released Updates
Finding updated packages
Downloading needed headers
Resolving dependencies
.Dependencies resolved
I will do the following:
[install: httpd 2.0.51-2.9.i386]
I will install/upgrade these to satisfy the dependencies:
[deps: apr 0.9.4-11.i386]
[deps: apr-util 0.9.4-14.2.i386]

Is this ok [y/N]: y
```

Yum client will match the newest Apache package available on repository server and then installs it. Two additional packages were installed for satisfying *httpd* package dependencies, those packages are part of APR(Apache Portable Runtime).

Creating "repository" web site

The server's default root location is "/var/www/html" directory, on index page Iuri will put a link to repository services and possible other messages such as Welcome messages and users notifications. The following HTML content will be configured as default index page and saved as *index.html* into web server's root:

```
<html>
<title>Proteus Web Server</title>
<div align="center">
<p>
<h1>Welcome, to access repository server packages click on link
below</h1>
<p>
<a href=/repository>RPM Repository</a>
</div>
</html>
```

Apache package comes with a file called *welcome.conf* located in "/etc/httpd/conf.d/" directory, Iuri has removed it, since there's already a index page configured as showed above. All below configurations will be done in Apache's main configuration file *httpd.conf*, in order to edit this file Iuri will execute the command "sudo vi /etc/httpd/conf/httpd.conf" as user *u401*:

For providing "/repository" target location, an *Alias* will be created pointing to repository server tree, where packages are located and a *Directory* entry for configuring access to this location. The following configuration settings will be included in */etc/httpd/conf/httpd.conf*:

```
# rpm repository
Alias /repository "/var/repository/"
<Directory "/var/repository">
        Options Indexes
        AllowOverride None
        Order deny,allow
        Allow from 192.168.1.0/24
        Deny from all
</Directory>
```

The directives[18] configured on *Directory* entry are:

18 "Apache core Features"
URL: http://httpd.apache.org/docs-2.0/mod/core.html (10 Oct. 2004)
"Apache Module mod_access"
URL: http://httpd.apache.org/docs-2.0/mod/mod_access.html (10 Oct. 2004)

- "Options Indexes": this option will allow users for viewing */var/repository* content, without having and index page configured on it;
- "AllowOverride None": this option disables any type of Apache's directive value declared after this line, thus any *.htaccess* file will be ignored;
- "Order deny,allow": this option controls the default access state and order in which *Allow* and *Deny* directives are evaluated;
- "Allow from 192.168.1.0/24": this option tells Apache to allow access from testing network IP addresses;
- "Deny from all": this option denies any host or IP address outside previous declared allowed range.

For starting the web server issue the following command:

```
[u401@proteus u401]$ sudo /sbin/service httpd start
```

The server was started but it complains about server's hostname, to set this name add the line below in *httpd.conf* file:

```
ServerName proteus
```

Another directive that will be activated on *httpd.conf* file is *IndexOptions*, this will control how files will be organized or indexed. The line below will suppress icons, html rules and last date modified information on listings and it will also make a version sort on file list:

```
IndexOptions SuppressIcon SuppressLastModified SuppressRules
VersionSort
```

After all above modifications, the web server can be restarted with the command "/sbin/service httpd restart" executed by *u401* user using SUDO interface. If everything runs fine, the service can be stopped since the hardening process was not applied yet. To stop the web server issue the command:

```
[u401@proteus u401]$ sudo /sbin/service httpd stop
```

Hardening Apache Web Server

The following steps will be taken by the administrator in order to minimize the chances of Apache service exploitation such as exploration of remote bugs, unused features or bad configurations and to restrict web services for only providing RPM packages download.

This action will be in accordance with checklist item "Network Services Security". The content of some configuration files including *httpd.conf* file will be described on Appendix B, at the end of this document.

1) Disabling unused modules and features

Fedora's Apache package comes with many modules[19] and features enable by default, so by disabling them the server becomes less vulnerable to remote attacks against useless enabled features. To disable a feature, place a "#" character at the beginning of the line. The command below lists the disabled Apache modules:

```
[u401@proteus u401]$ grep ^#LoadModule /etc/httpd/conf/httpd.conf
#LoadModule auth_module modules/mod_auth.so
#LoadModule auth_anon_module modules/mod_auth_anon.so
#LoadModule auth_dbm_module modules/mod_auth_dbm.so
#LoadModule auth_digest_module modules/mod_auth_digest.so
#LoadModule ldap_module modules/mod_ldap.so
#LoadModule auth_ldap_module modules/mod_auth_ldap.so
#LoadModule env_module modules/mod_env.so
#LoadModule dav_module modules/mod_dav.so
#LoadModule status_module modules/mod_status.so
#LoadModule asis_module modules/mod_asis.so
#LoadModule info_module modules/mod_info.so
#LoadModule dav_fs_module modules/mod_dav_fs.so
#LoadModule vhost_alias_module modules/mod_vhost_alias.so
#LoadModule imap_module modules/mod_imap.so
#LoadModule actions_module modules/mod_actions.so
#LoadModule speling_module modules/mod_speling.so
#LoadModule userdir_module modules/mod_userdir.so
#LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
#LoadModule proxy_http_module modules/mod_proxy_http.so
#LoadModule proxy_connect_module modules/mod_proxy_connect.so
#LoadModule suexec_module modules/mod_suexec.so
#LoadModule cgi_module modules/mod_cgi.so
```

2) Disabling manual pages

After disabling modules, all default directories that are not going to be used are removed and disabled in configuration file. This will minimize the chances of server information discovery such as Apache's version. Iuri removes all entries that make references to "manual" word as showed below:

```
#AliasMatch ^/manual(?:/(?:de|en|fr|ja|ko))?(/.*)?$ "/var/www/manual$1"

#<Directory "/var/www/manual">
#    Options Indexes
#    AllowOverride None
#    Order allow,deny
#    Allow from all
#
#    <Files *.html>
#        SetHandler type-map
```

19 "Apache Module Index" URL: http://httpd.apache.org/docs-2.0/mod/ (10 Oct. 2004)

```
#    </Files>
#
#    SetEnvIf Request_URI ^/manual/de/ prefer-language=de
#    SetEnvIf Request_URI ^/manual/en/ prefer-language=en
#    SetEnvIf Request_URI ^/manual/fr/ prefer-language=fr
#    SetEnvIf Request_URI ^/manual/ja/ prefer-language=ja
#    SetEnvIf Request_URI ^/manual/ko/ prefer-language=ko
#    SetEnvIf Request_URI ^/manual/ru/ prefer-language=ru
#    RedirectMatch 301 ^/manual(?:/(de|en|fr)){2,}(/.*)?$ /manual/$1$2
#</Directory>
```

NOTE: Apache's manual files are found on *httpd-manual* RPM package, which is not installed on repository server.

3) Disabling unused *AddHandler* directives

This option enables certain file extensions mapping to "handlers", if a handler is activated and an attacker has rights to put a file with any extension on web tree, he can make the file executable and maps the file to a handler such as CGI script. To disable that feature simple comment out lines beginning with this option, the following lines were disabled on repository server:

```
[u401@proteus u401]$ grep "^[[:blank:]]*#AddHandler " /
etc/httpd/conf/httpd.conf
#AddHandler cgi-script .cgi
#AddHandler send-as-is asis
```

4) Disabling Server's signature

By default the server displays version information on certain page's footer such as error pages, this is valuable information for attackers. To disable this function issue the line below on configuration file:

```
ServerSignature Off
```

5) Disabling icons access

Icons won't be used on repository server, some default icons reveal information about server's version, thus icons access is disabled:

```
#Alias /icons/ "/var/www/icons/"

#<Directory "/var/www/icons">
#    Options Indexes MultiViews
#    AllowOverride None
#    Order allow,deny
#    Allow from all
#</Directory>
```

6) Disabling CGI support

CGI support is not necessary on repository server, since its content will not be executed, if any attacker get access to cgi enable directory he won't be able to execute scripts or programs. To disable dynamic processing through CGI scripts

comment out any *ScriptAlias* directive and the corresponding *Directory* entry.

```
#ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
#<Directory "/var/www/cgi-bin">
#    AllowOverride None
#    Options None
#    Order allow,deny
#    Allow from all
#</Directory>
```

7) Disabling server's header info about additional modules or components

On default configuration Apache returns additional information about modules and components installed such as scripting language modules(mod_php, mod_perl), to disable this feature put the following line in configuration file:

```
ServerTokens Prod
```

This action will make web server displaying only “Apache” string on server's http responses, minimizing the chances of server's information discovery.

8) Disabling unused HTTP methods

Default enabled methods include POST, TRACE and OPTIONS, the only allowed will be GET and HEAD, in order to make this configuration put the following lines on configuration file:

```
RewriteEngine On
RewriteCond %{REQUEST_METHOD} ^(TRACE|POST|OPTIONS)
RewriteRule .* - [F]
```

9) Disabling unused *FollowSymLink* and *Indexes* directives

FollowSymLink directive enable web users to access files symbolic linked, this can lead to symlink attacks and *Indexes* directive can exposes sensitive data when an index page is not present. To disable them, remove all these directive entries or add a minus sign in front of all directives by those names. Below is the configuration applied to "/var/www/html" *Directory* entry:

```
Options -Indexes -FollowSymLinks
```

NOTE: The final content of httpd.conf configuration file excluding comments and blank lines will be described on Appendix B.

There are also some features and configurations to be verified, but on modern Apache distributions packages, many of them come disabled by default such as “.htaccess” and “.htpasswd” files protection:

```
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
```

After all hardening steps taken, Iuri must start the server by issuing the

following command on the shell:

```
[u401@proteus u401]$ sudo /sbin/service httpd start
```

At this point the web server is ready for providing packages download for network users using yum interface, next step is about the network clients configuration.

Network Users Operating System Upgrading using YUM

The server is now ready to provide RPM packages for Linux users of GIAC Enterprises testing environment. Iuri has provided the following *yum.conf* file for users, it must be copied to all Linux machines, the right location is in */etc* directory. This action requires superuser access since */etc* directory is only writable by *root* user.

The same repository entries are defined: *os*, *plus* and *updates*. But now pointing to a web server, repository's Apache server:

```
# yum.conf on network clients
[main]
cachedir=/var/cache/yum
debuglevel=2
logfile=/var/log/yum.log
pkgpolicy=newest
distroverpkg=redhat-release
tolerant=1
exactarch=1
retries=20

[base]
name=Fedora Core $releasever - $basearch – Base
baseurl=http://192.168.1.200/repository/linux/fedora/$releasever/os/


[updates-released]
name=Fedora Core $releasever - $basearch - Released Updates
baseurl=http://192.168.1.200/repository/linux/fedora/$releasever/updates/


[plus]
name=Fedora Core $releasever - $basearch – Extra Packages
baseurl=http://192.168.1.200/repository/linux/fedora/$releasever/plus/
```

The same yum update command can now be executed on any Linux client holding the above configuration file. The final *yum.conf* file for both the server and clients will be described in Appendix B.

YUM Interface Commands

Now, users are able to execute the following *yum* commands for RPM packages management on their machines using repository server solution, these commands are useful since no more local medias will be necessary for managing RPM database.

Main yum interface commands, more information can be found on *yum* command manual page:

- yum **update** [*package*|*list*] : this update *package* or a list of packages, the hole system is updated if no parameter is provided
- yum **install** [*package*|*list*] : installs *package* or a list of packages
- yum **remove** [*package*|*list*] : remove *package* or a list of packages
- yum **list** [*regex string*] : find packages matching *regex string* on all rpm database repositories including the local database(installed packages).

Hardening OpenSSH Server

When the server becomes completely ready, it will be located on GIAC Enterprises data center, thus Iuri will need remote connection to the server since it won't be interesting entering machines room frequently.

OpenSSH server package is installed by default and provides for remote console access. The following steps will be in accordance with checklist item "Network Services Security".

The main OpenSSH server configuration file is */etc/ssh/sshd_config*, so all modifications have to be done at this file. Superuser rights are required for editing that file, thus it will be executed by *u401* user using SUDO command and *vi* editor.

```
[u401@proteus u401]$ sudo vi /etc/ssh/sshd_config
```

1) Protocol 2 restriction

Iuri will only use Protocol 2 features, so the first version can be disabled. The following line restricts the utilization of only 2 version protocol on the server:

```
Protocol 2
```

2) Listening IP address restriction

For testing environment, only 192.168.1.200 IP address will be listening for incoming SSH connections. If someone got privileges for loading additional IP address on the server, SSH port won't be binded to that new IP address. Configuration follows:

```
ListenAddress 192.168.1.200
```

3) Disabling Superuser login

Superuser's environment must be accessible only by administrative group members, allowing directly *root* login access will not be in accordance with this security practice, any person who get the password will become able to remotely take control over the server.

Superuser remote login is enabled by default, to disable this “feature” the following configuration line is needed:

```
PermitRootLogin no
```

4) User login and client IP restriction

Only *u401* user will be able to login using SSH network service, any other user will not be allowed. If an attacker successfully exploit any other account, he will be able to remote login only by changing server's configuration file or by using *u401* user.

Iuri's machine IP address on testing environment is 192.168.1.48, this will be the only machine used by him for administrating repository server. For applying these definitions, the following line must be added to *sshd_config* file:

```
AllowUsers u401@192.168.1.48
```

5) Authentication warning message

Showing a warning message before authentication dialog is important for getting legal protection against possible security incidents. The content of this message will be located in */etc/issue.net* file. Iuri has executed the following command to populate the file containing the message:

```
[u401@proteus /]$ sudo sh -c "echo 'Access restricted to authorized
users only' > /etc/issue.net"
```

For enabling the above warning message on SSH server put the following line in configuration file:

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

6) Disabling *Subsystems*

OpenSSH package comes with SFTP daemon feature enabled by default, so this daemon is automatically started when the server comes up. Iuri won't use this feature, thus disabling any line starting with “Subsystem” word:

```
#Subsystem      sftp    /usr/libexec/openssh/sftp-server
```

As it happens with Apache configuration, some security settings are already right configured, these are default settings. The following list is related to default OpenSSH settings related to server's security:

- Rhosts settings: The server can read configurations from *~.rhosts* files, this kind of authentication is insecure and must be disabled, the

parameters that come already right configured are:
RhostsAuthentication no
IgnoreRhosts yes
RhostsRSAAuthentication no

- Strict Modes: To force SSH server to check configuration files modes and ownership, the parameter *StrictModes* must be set to *yes*;
- Blank passwords: The parameter *PermitEmptyPasswords* must be set to *no* for disabling empty passwords;
- Privileges Separation: The default value for *UsePrivilegeSeparation* parameter is *yes*, this configuration prevents privilege escalation attempts by using user privileges for creating authentication process;

Operating System Security

After network services(Apache and SSH servers) implementation and hardening, Iuri now have to apply security settings on Fedora Linux operating system, this action will be in accordance with checklist item "Operating System Security". After this stage, some auditing process will be taken for validate security configurations. The following steps will be implemented:

1) Network security using *sysctl* kernel parameter interface

The following kernel parameters will be added to */etc/sysctl.conf* file:

net.ipv4.conf.all.accept_source_route=0 this parameter disables source route feature on pre-routed packets;
net.ipv4.conf.all.accept_redirects=0 this parameter disables packets redirecting;
net.ipv4.conf.all.log_martians=1 this parameter logs all unknown incoming packets that arrives on server's interface;
net.ipv4.conf.all.rp_filter=1 this will parameter enable spoofing protection on server's interface;
net.ipv4.icmp_echo_ignore_broadcasts=1 this parameter will force the server to ignore icmp broadcasts for avoiding denial of service;
net.ipv4.icmp_ignore_bogus_error_responses=1 this parameter will ignore warning icmp messages for avoiding log filing;
net.ipv4.tcp_syncookies=1 this parameter enables protection against syn flood attacks avoiding denial of service.

Note that the parameter *net.ipv4.ip_forward* which controls packets forwarding comes disabled by default. After editing *sysctl.conf* configuration file, the following command will enable modifications:

```
[u401@proteus u401]$ sudo /sbin/sysctl -p
```

2) Correct log files modes and rotation

Sensitive information must be accessible only by root or administrative group members, thus 0640 mode will be applied for some important log information residing on */var/log* directory files as showed in commands below:

```
[u401@proteus log]$ sudo chmod 640 dmesg yum.log
```

Logrotate program is responsible for rotating log files on Fedora Linux, including *syslog* files which content is very important, *logrotate.d* directory and *logrotate.conf* file are the files that hold *logrotate* configuration.

The following content correspond to correct */etc/logrotate.d/syslog* file:

```
/var/log/messages /var/log/secure /var/log/maillog /var/log/spooler /
var/log/boot.log /var/log/cron {
    create mode 0640
    sharedscripts
    postrotate
        /bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /
dev/null || true
    endscript
}
```

The same line in bold showed above have to be added to new log files entries that store sensitive information. The content of logrotate.conf file will be described on Appendix B, no command need to be executed seeing that *logrotate* program is a *cron* job.

3) Password Policy Application

Password expiration time and maximum length values must be set for avoiding password repetition and short passwords. The following lines must exist on configuration file */etc/login.defs*:

```
# PASS_MAX_DAYS   Maximum number of days a password may be used.
# PASS_MIN_DAYS   Minimum number of days allowed between password
changes.
# PASS_MIN_LEN    Minimum acceptable password length.
# PASS_WARN_AGE   Number of days warning given before a password
expires.

PASS_MAX_DAYS   30
PASS_MIN_DAYS   20
PASS_MIN_LEN    12
PASS_WARN_AGE   7
```

NOTE: The users *u300* and *u401* were created before password policy application, so */etc/shadow* file, which holds password policy data, must be upgraded. For accomplish this task, issue the commands:

```
[u401@proteus etc]$ sudo chage -m 20 -M 30 -W 7 u300
[u401@proteus etc]$ sudo chage -m 20 -M 30 -W 7 u401
```

To validate these actions the following commands can be executed to list each user active settings:

```
[u401@proteus etc]$ sudo chage -l u300
[u401@proteus etc]$ sudo chage -l u401
```

4) Banner files content

The files *motd* and *issue* must show a waring message, as done with *issue.net* file already modified on SSH server hardening process. Showing a warning message for users is important for getting legal protection against possible security incidents.

The following content will be added to files /*etc/motd* and */etc/issue*:

```
Access restricted to authorized users only
```

5) SUID/SGID bit removal

SUID and SGID bit controls file execution policy, where users can execute commands using another user or group privileges. When SUID bit is enabled on an executable and this file has *root* user ownership, if a user can execute it, during program execution the user will earn superuser's privileges. The same feature applies for SGID bit, but now involving group ownership privileges.

Files with those bits enabled that won't be used must have their modes changed with *chmod* command. The following commands will list SUID/SGID enable files on the server and remove the those bits from them:

```
[u401@proteus etc]$ sudo find / -perm +6000 -type f 2>/dev/null
[u401@proteus etc]$ sudo find / -perm +6000 -type f 2>/dev/null | grep
-v "/bin/su\|pam\|chk\|cron\|bin/at\|sudo\|passwd\|lock\|utempter\|
user" | xargs -i sudo chmod -s {}
```

The values passed to *grep* program will not be listed on output to *xargs* program, so those values are the list of programs that will keep their SUID/SGID bits enabled.

6) Reboot keyboard shortcut removal

Linux comes with “ctrl+alt+del” keyboard shortcut enabled by default, any user with physical access becomes able to reboot the server by pressing these keys. To disable this feature insert a “#” symbol on the line related to this configuration in */etc/inittab* file:

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

7) Shell Timeout configuration

For enabling shell timeout, the following variable must be defined on */etc/profile* configuration file:

```
TMOUT=300
```

This action will force shell logout after 5 minutes or 300 seconds of maximum idle time.

8) Boot configuration about network services and applications

Some services must be automatically enabled after system boot and useless programs must be disabled. On Fedora Linux this can be configured with *ntsysv* or *chkconfig* programs. Apache web server needs to be loaded automatically, netfs and rhnsd applications must be disabled, so the following commands will make such tasks:

```
[u401@proteus u401]$ sudo /sbin/chkconfig --level 35 httpd on
[u401@proteus u401]$ sudo /sbin/chkconfig --level 35 netfs off
[u401@proteus u401]$ sudo /sbin/chkconfig --level 35 rhnsd off
```

The following list is the output of services and applications that will be enable or disabled after system boot, the same list can be taken by executing *ntsysv* command, names in bold are main enabled repository server applications:

```
[u401@proteus u401]$ sudo /sbin/chkconfig –list
atd             0:off   1:off   2:off   3:on    4:on    5:on    6:off
sshd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
readahead_early 0:off   1:off   2:off   3:off   4:off   5:on    6:off
random          0:off   1:off   2:on    3:on    4:on    5:on    6:off
microcode_ctl   0:off   1:off   2:off   3:on    4:on    5:on    6:off
netplugd        0:off   1:off   2:off   3:off   4:off   5:off   6:off
netfs           0:off   1:off   2:off   3:off   4:on    5:off   6:off
rawdevices      0:off   1:off   2:off   3:on    4:on    5:on    6:off
kudzu           0:off   1:off   2:off   3:on    4:on    5:on    6:off
gpm             0:off   1:off   2:on    3:on    4:on    5:on    6:off
rhnsd           0:off   1:off   2:off   3:off   4:on    5:off   6:off
saslauthd       0:off   1:off   2:off   3:off   4:off   5:off   6:off
netdump         0:off   1:off   2:off   3:off   4:off   5:off   6:off
httpd           0:off   1:off   2:off   3:on    4:off   5:on    6:off
psacct          0:off   1:off   2:off   3:off   4:off   5:off   6:off
irda            0:off   1:off   2:off   3:off   4:off   5:off   6:off
iptables        0:off   1:off   2:on    3:on    4:on    5:on    6:off
cpuspeed        0:off   1:on    2:on    3:on    4:on    5:on    6:off
smartd          0:off   1:off   2:on    3:on    4:on    5:on    6:off
yum             0:off   1:off   2:off   3:off   4:off   5:off   6:off
syslog          0:off   1:off   2:on    3:on    4:on    5:on    6:off
apmd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
crond           0:off   1:off   2:on    3:on    4:on    5:on    6:off
anacron         0:off   1:off   2:on    3:on    4:on    5:on    6:off
acpid           0:off   1:off   2:off   3:on    4:on    5:on    6:off
```

```
readahead       0:off   1:off   2:off   3:off   4:off   5:on    6:off
irqbalance      0:off   1:off   2:off   3:on    4:on    5:on    6:off
network         0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

9) Firewall Rules implementation

The following services will be running on repository server:

- WWW / Port 80 / TCP, through Proxy Server / Port 3128
- SSH / Port 22 / TCP

The following services need to be accessed from the server:

- WWW / Port 80 / TCP
- DNS / Port 53 / UDP

Starting from these information, Iuri knows that only INPUT and OUTPUT chains will be used, no NAT or FORWARD rules will be needed. The following firewall rules will be listed on */etc/sysconfig/iptables* file, note that each rule has its comments, which explain the main reason for its existence:

```
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]

# rules
# loopback traffic allowed
-A INPUT -i lo -j ACCEPT

# packets about an already established connection, the return
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# incoming services
-A INPUT -p tcp -s 192.168.1.0/24 -m multiport --dports 22,80 -j ACCEPT

# rejecting other packets from internal users will be better than DROP
# for some applications functioning
-A INPUT -s 192.168.1.0/24 -j REJECT

# www through proxy server
-A OUTPUT -p tcp -d 192.168.1.12 --dport 3128 -j ACCEPT

# name servers
-A OUTPUT -p udp -d 192.168.1.24 --dport 53 -j ACCEPT
-A OUTPUT -p udp -d 192.168.1.29 --dport 53 -j ACCEPT

# end

COMMIT
```

To activate firewall rules issue the command below:

```
[u401@proteus u401]$ sudo /sbin/service iptables restart
```

Server's Auditing Process

There are important steps to be taken for validating Iuri's implementation, some of these steps will be done remotely, outside server's machine. The following actions will be in accordance with checklist item "Operating System and Services Auditing". Also note that many auditing steps or validations were done right after some implementation steps.

Network Scanning

In order to list open UDP and TCP ports on repository server, Iuri has installed some useful software on his machine(192.168.1.48) including *nmap*[20] network tool, this will be used to accomplish the task of port scanning.

The following commands executed on Iuri's machine will list open UDP and TCP ports on repository server:

```
# nmap -sU -n -P0 192.168.1.200

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-12-14
03:45 BRST

All 1659 scanned ports on 192.168.1.200 are: closed

# nmap -sT -n -P0 192.168.1.200

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-12-14
03:48 BRST

Interesting ports on 192.168.1.200:

(The 1658 ports scanned but not shown below are in state: closed)

PORT   STATE SERVICE

22/tcp open  ssh

80/tcp open  http
```

As listed above, only two ports are open: 22 and 80. These are the only network services running on the server.

OpenSSH Server Auditing

To validate SSH daemon security configurations Iuri has made some tests:

- Login attempt from other machine, with different IP address than his machine's one;
- From his machine, which IP address is allowed, he has tried to login with *root*'s account and with *u300* user;
- From his machine, he has tried to use SFTP feature;

20 Nmap ("Network Mapper") is a free open source utility for network exploration or security auditing. URL: http://www.insecure.org/nmap/ (14 Oct. 2004)

All above tests have failed because server settings are configured to allow connections from only Iuri's machine IP address, *u401* user and only SCP file transfer method.

Apache Server Auditing

Nikto is an Open Source(GPL) web server scanner which performs comprehensive tests against web servers for multiple items, including over 3100 potentially dangerous files/CGIs, versions on over 625 servers, and version specific problems on over 230 servers[21].

Iuri has installed this tool on his computer for checking repository implementation. Below are the results of running the tool against repository server, note that no potential bugs or bad configurations were found validating Apache hardening steps taken.

```
$ ./nikto.pl -nolookup -host 192.168.1.200
---------------------------------------------------------------------------
- Nikto 1.34/1.29     -     www.cirt.net
+ Target IP:        192.168.1.200
+ Target Hostname: 192.168.1.200
+ Target Port:     80
+ Start Time:      Tue Dec 14 04:21:25 2004
---------------------------------------------------------------------------
- Scan is dependent on "Server" string which can be faked, use -g to
override
+ Server: Apache
- Server did not understand HTTP 1.1, switching to HTTP 1.0
+ Server does not respond with '404' for error messages (uses '400').
+     This may increase false-positives.
+ No CGI Directories found (use '-C all' to force check all possible
dirs)
+ 1833 items checked - 0 item(s) found on remote host(s)
+ End Time:        Tue Dec 14 04:21:38 2004 (13 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Note that server string is “Apache” only, no modules or other server information are exposed. Additionally Iuri has generated page errors by accessing pages that don't exist in order to validate server signature hiding(default footer), he also has browsed index page for validating indexing and icons hiding features.

21 URL: http://www.cirt.net/code/nikto.shtml (14 Oct. 2004)

Operating System Auditing

The main test is about software updating, Iuri must know if all security patches are right applied and stored on the server, this action can be accomplished by listing latest update packages released on Fedora's website and then checking their availability on local repository server which stores RPM packages in */var/repository/linux/fedora/2/updates* directory. Iuri can also check for those packages entries on yum_repository script logs.

After checking that all packages are stored on local updates repository, in order update all server software the following command must be executed:

```
[u401@proteus u401]$ sudo yum update
Password:
Gathering header information file(s) from server(s)
Server: Fedora Core 2 - i386 – Base
Server: Fedora Core 2 - i386 - Released Updates
Finding updated packages
Downloading needed headers
No Packages Available for Update
No actions to take
```

As seen above no more packages are available for upgrading, the same “yum update” command can be executed on other network machine for checking repository files availability through the network and complete the task of validating main implementation focus.

At this point Iuri now has all documented steps for implementing a secure repository solution on production environment.

Final Considerations

Manual or Automatic Updates

Yum update command is very simple and can be called without interactivity by passing "-y" parameter, this allows administrators to start update process as a *cron* job. The advantage of this decision is that as soon as a package becomes available on repository server it will be deployed the machine without administrator intervention.

There are also some issues about this and that why Iuri has decided to manually start update command on his servers: new software needs to be homologated, without testing it they can become a problem; machines could not have necessary resources for some update action such as disk space.

Features that could be added for repository security

Fedora Linux 2 and future versions are shipped with Security-enhanced Linux(SELinux), which is a research prototype of the Linux kernel and a number of utilities with enhanced security functionality designed simply to demonstrate the value of mandatory access controls to the Linux community and how such controls could be added to Linux[22]. It's a complex subject that needs to be deeply studied tested, thus it is not on this document's scope.

22 URL: http://www.nsa.gov/selinux/info/faq.cfm#I1 (20 Oct. 2004)

References

SANS Books

Koconis, David; Murray, Jim; Purvis, Jos; Wassom, Darrin. Securing Linux. A Survival Guide for Linux(Version 1.0). SANS Press, February 2003.

SANS Practical Papers

Murdoch, Don. “Building a Secured OS for a Root Certificate Authority”. URL: http://www.giac.org/practical/GCUX/Don_Murdoch_GCUX.pdf (24 Nov. 2004).

Heilman, Marshall. “Implementing a Shorewall Firewall and BIND DNS Server on a Hardened Fedora Core 2 OS”, Aug 2, 2004. URL: http://www.giac.org/practical/GCUX/Marshall_Heilman_GCUX.pdf (24 Nov. 2004).

Wald, Rickey. “Securing Red Hat Linux 9 as an Apache Web Server, VSFTP Server and MySQL Server”, Dec 5, 2003. URL: http://www.giac.org/practical/GCUX/Ricky_Wald_GCUX.pdf (24 Nov. 2004).

Lam, Jason. “Securing MySQL Server on FreeBSD 4.5”. URL: http://www.giac.org/practical/Jason_Lam_GCUX2.pdf (24 Nov. 2004).

Websites

University of New Castle, “Writing Research Theses or Dissertations”, URL: http://lorien.ncl.ac.uk/ming/Dept/Tips/writing/thesis/thesis-cite.htm (29 Oct. 2004).

YUM Website, URL: http://www.linux.duke.edu/projects/yum/ (29 Aug. 2004).

RPM Package Manager Website, URL: http://www.rpm.org (29 Oct. 2004).

Fedora Linux Project Website, URL: http://fedora.redhat.com (29 Oct. 2004).

FedoraNEWS Website, URL: http://www.fedoranews.org (22 Nov. 2004).

SUDO Website, URL: http://www.courtesan.com/sudo/ (18 Sep. 2004).

Red Hat Linux On-line Documentation, URL: http://www.redhat.com/docs/ (18 Sep. 2004).

Apache 2.0 On-line Documentation, URL: http://httpd.apache.org/docs-2.0/ (10 Oct. 2004)

The Linux Documentation Project, URL: http://www.tldp.org/ (18 Nov. 2004).

RPM Search Database, URL: http://rpm.pbone.net/ (12 Oct. 2004).

Appendix A: Packages List

This is the final complete list of installed software on repository server machine, it was obtained by issuing the command “rpm -qa | sort | column”:

```
acl-2.2.7-5                                  lvm2-2.00.15-2
acpid-1.0.2-6                                mailcap-2.1.15-1
anacron-2.3-30                               mailx-8.1.1-32
apmd-3.0.2-22                                MAKEDEV-3.3.13-1
apr-0.9.4-11                                 man-1.5o1-6
apr-util-0.9.4-14.2                          man-pages-1.66-2
ash-0.3.8-18                                 mingetty-1.07-2
aspell-0.50.3-19.1                           mkinitrd-3.5.22-1
aspell-en-0.51-7.1                           mktemp-1.5-7
at-3.1.8-53                                  modutils-2.4.26-16
attr-2.4.1-4                                 mtr-0.54-5
authconfig-4.6.2-1                           mt-st-0.7-13.1
basesystem-8.0-3                             nano-1.2.3-1
bash-2.05b-38                                ncurses-5.4-5
bc-1.06-16.1                                 netconfig-0.8.20-1.1.1
beecrypt-3.1.0-3                             netdump-0.6.9-3.1
bind-libs-9.2.3-13                           net-tools-1.60-25.1
bind-utils-9.2.3-13                          newt-0.51.6-2.1.1
bzip2-1.0.2-12.1                             ntsysv-1.3.9-1.1
bzip2-libs-1.0.2-12.1                        openldap-2.1.29-1
chkconfig-1.3.9-1.1                          openssh-3.6.1p2-34
comps-2-0.20040513                         openssh-clients-3.6.1p2-34
coreutils-5.2.1-7                           openssh-server-3.6.1p2-34
cpio-2.5-6                                   openssl-0.9.7a-35
cracklib-2.7-27.1                            pam-0.77-40
cracklib-dicts-2.7-27.1                      pam_krb5-2.0.10-1
crontabs-1.10-6                              parted-1.6.9-3
cyrus-sasl-2.1.18-2.2                        passwd-0.68-8.1
cyrus-sasl-md5-2.1.18-2.2                    pax-3.0-8
cyrus-sasl-plain-2.1.18-2.2                 pciutils-2.1.99.test3-1.1
db4-4.2.52-3.1                               pcre-4.5-2
dev-3.3.13-1                                 perl-5.8.3-18
device-mapper-1.00.14-3                      perl-Filter-1.30-5
devlabel-0.42.05-3.1                         pinfo-0.6.8-4
diffutils-2.8.1-11                           policy-1.11.3-3
dump-0.4b33-3                                policycoreutils-1.11-2
e2fsprogs-1.35-7.1                           popt-1.9.1-0.3
eject-2.0.13-5                               portmap-4.0-59
elfutils-0.95-2                              prelink-0.3.2-1
elfutils-libelf-0.95-2                       procmail-3.22-13
ethtool-1.8-3.1                              procps-3.2.0-1.1
fbset-2.1-15                                 psacct-6.3.2-29
fedora-logos-1.1.24-1                        psmisc-21.4-2
fedora-release-2-4                           pyOpenSSL-0.5.1-21.1
file-4.07-4                                  python-2.3.3-6
```

```
filesystem-2.2.4-1                                  python-optik-1.4.1-5
findutils-4.1.7-25                                  pyxf86config-0.3.18-2
freetype-2.1.7-4                                    raidtools-1.00.3-8
gawk-3.1.3-7                                        rdate-1.3-3.1
gdbm-1.8.0-22.1                                     readline-4.3-10.1
glib-1.2.10-12.1.1                                  rhnlib-1.5-1.1
glib2-2.4.7-1.1                                     rhpl-0.143-1
glibc-2.3.3-27.1                                    rmt-0.4b33-3
glibc-common-2.3.3-27.1                             rootfiles-7.2-7
gmp-4.1.2-14                                        rpm-4.3.1-0.3
gnupg-1.2.4-2.1                                     rpm-python-4.3.1-0.3
gpg-pubkey-4f2a6fd2-3f9d9d3b                        schedutils-1.3.0-6
gpg-pubkey-6b8d79e6-3f49313d                        sed-4.0.8-4
gpm-1.20.1-49                                       setarch-1.4-1
grep-2.5.1-26                                       setserial-2.17-15
groff-1.18.1-34                                     setup-2.5.33-1
grub-0.94-5                                         setuptool-1.15-1
gzip-1.3.3-12                                       shadow-utils-4.0.3-21
hdparm-5.5-1                                        slang-1.4.9-12
hesiod-3.0.2-29.1                                   slocate-2.7-9
hotplug-2004_04_01-1                                specspo-9.0.92-1.1
httpd-2.0.51-2.9                                    star-1.5a25-5
hwdata-0.120-1                                      statserial-1.1-34
info-4.7-4                                          sudo-1.6.7p5-26
initscripts-7.55.1-1                                symlinks-1.2-21
iproute-2.4.7-14                                    sysklogd-1.4.1-16
iptables-1.2.9-2.3.1                          system-config-mouse-1.2.6-2
iputils-20020927-13          system-config-network-tui-1.3.17-0.FC2.1
irda-utils-0.9.15-5          system-config-securitylevel-tui-1.3.12-1
kbd-1.12-1                                          SysVinit-2.85-25
kernel-2.6.9-1.3_FC2                                tar-1.13.25-14
kernel-utils-2.4-9.1.131                            tcpdump-3.8.2-6.FC2.1
krb5-libs-1.3.4-6                                   tcp_wrappers-7.6-36
krbafs-1.2.2-2.1                                    termcap-11.0.1-18.1
kudzu-1.1.68.2-1                                    time-1.7-24
less-382-3                                          tmpwatch-2.9.0-2.1
lftp-2.6.12-1                                       tzdata-2004e-1.fc2
lha-1.14i-14.1                                      unix2dos-2.2-21
libacl-2.2.7-5                                      unzip-5.50-37
libattr-2.4.1-4                                     up2date-4.3.19-1
libgcc-3.3.3-7                                      usbutils-0.11-4
libselinux-1.11.4-1                                 usermode-1.70-2
libstdc++-3.3.3-7                                   utempter-0.5.5-4
libtermcap-2.0.8-38                                 util-linux-2.12-18
libuser-0.52.5-0.FC2.1                              vconfig-1.8-2
libwvstreams-3.70-13.1                              vim-minimal-6.2.457-1
libxml2-2.6.16-2                                    vixie-cron-3.0.1-87
libxml2-python-2.6.16-2                             wget-1.9.1-16.fc2
lockdev-1.0.1-2.3.1                                 which-2.16-2
logrotate-3.7-4.1                                   words-2-22
```

```
logwatch-5.1-3                                  yum-2.0.7-1.1
lrzsz-0.12.20-18                                zlib-1.2.1.1-2.1
lsof-4.68-2
```

Appendix B: Main Script and Configuration Files Used

The following content describes main scripts and configuration files used on repository implementation, note that some IP addresses within those files are based on testing environment of GIAC Enterprises, which has 192.168.1.0/24 network IP address.

Custom Script: *yum_repository.sh*

```
#!/bin/sh

# Wgets(http) rpm packages then executes yum-arch
# creating the repository headers on local machine
# http://fedoranews.org/alex/tutorial/yum/
# Last Modified: 15/11/2004

# Setup including:
# Mirror examples for RH9 and Fedora Base/Updates
# Yum.conf configuration
# GPG/MD5 checking, Proxy support
# yum-arch and createrepo compatible

# Visit FedoraNEWS Website http://fedoranews.org
# Author: Alexandre de Abreu [alex@fedoranews.org]

# For the impatient ones: just execute this script
# and follow the advice of the error messages

# Fill the MIRROR variables as you wish and test running
# on the shell, after things working try the line below
# to setup a cron job for executing every six hours:
# 1 */6 * * * user /path/to/yum_repository.sh

# Configure your clients /etc/yum.conf with the following
# lines and start a web/ftp server on the repository
# [updates-local]
# name=Linux $releasever - $basearch – Updates
# baseurl=http://repository_ipaddr/path/to/repository/
# OR
# baseurl=ftp://repository_ipaddr/path/to/repository/

# And change the local repository server yum.conf to:
# baseurl=file:///path/to/repository/

# Mirrors arrays  ###################################
# Try to use those with "Indexes" Apache option enabled
# Fedora mirrors http://fedora.redhat.com/download/mirrors.html
# RedHat mirrors http://www.redhat.com/download/mirror.html
# Yum RPM for Red Hat
http://www.linux.duke.edu/projects/yum/download.ptml
```

```
# Put any numbers of mirrors here, sequentially:
# MIRROR_URL[X]="http://url"
# MIRROR_DIR[X]="/filesystem/path"
# MIRROR_URL    -> Where to get .rpm files, must be a URL
# MIRROR_DIR    -> Where .rpm files and repository struct will be on
the disk
# X             -> Subscript, array index, must begin with 0

# Fedora 3 Updates Mirror

MIRROR_URL[0]
="http://distro.ibiblio.org/pub/linux/distributions/fedora/linux/core/u
pdates/2/"

MIRROR_DIR[0]="/var/repository/linux/fedora/2/updates/"

# Fedora 2 Updates Mirror

#MIRROR_URL[1]
="http://distro.ibiblio.org/pub/linux/distributions/fedora/linux/core/u
pdates/2/"

#MIRROR_DIR[1]="/var/ftp/pub/linux/fedora/2/updates/"

# Red Hat Updates Mirror

#MIRROR_URL[0]
="http://distro.ibiblio.org/pub/linux/distributions/redhat/updates/9/en
/os/"

#MIRROR_DIR[0]="/var/ftp/pub/linux/redhat/9/updates/"

# The following can be disabled after download completes
# Points to rpm packages that come with installation CD/ISO
# Fedora Core 1 and Red Hat 9 tooks 3.6G of HD space
# After this working, you can forget CD medias and just do
# yum install package and stuff, see this article for more:
# http://fedoranews.org/tchung/howto/2003-11-09-yum-intro.shtml

# Fedora Base Mirror

# MIRROR_URL[2]
="http://rpmfind.net/linux/fedora/core/1/i386/os/Fedora/RPMS/"

# MIRROR_DIR[2]="/var/ftp/pub/linux/fedora/1/os/"

# Red Hat Base Mirror
# MIRROR_URL[3]
="http://rpmfind.net/linux/redhat/9/en/os/i386/RedHat/RPMS/"
# MIRROR_DIR[3]="/var/ftp/pub/linux/redhat/9/os/"

# If you want HTTP Proxy support, fill at least
# PROXY_SERVER and PROXY_PORT variables
# To disable proxy support left it blank
PROXY_SERVER="192.168.1.12"
PROXY_PORT="3128"

# Be carefull with the following credentials, any "ps" will show them
# Use a public user or create rules w/o auth just from this machine
```

```
# when connecting to Mirror's IP addresses tcp port 80(http)
PROXY_USER=""
PROXY_PASS=''

########################################################
# Do not edit below this line unless you know what
# you are doing

# Filter what is intersting for us
IGNORE_FILES="*-debuginfo-*,*\.src\.rpm,*\.hdr"
ALLOW_FILES="*i[356]86\.rpm,*athlon\.rpm,*noarch\.rpm"

# Default umask
DEF_UMASK=003

# Log file, where all output will go
# If you dont want logging set LOG_FILE to /dev/null
# If you want to rotate it with logrotate every 1Mb
# edit /etc/logrotate.conf and add the following
# /tmp/yum_repository.log {
#        compress
#        nomail
#        missingok
#        notifempty
#        rotate 2
#        size 2M
# }
LOG_FILE="/var/log/yum_repository.log"

# Allow resume[-c]
# Do not create domain dir[-nH]
# Do not go to parent dirs[-np]
# Be recursive, needed[-r]
# Append to output log[-a]
WGET_ARGS="-a $LOG_FILE -R $IGNORE_FILES -A $ALLOW_FILES -np -nH -c -r"

WGET=$(/usr/bin/which --skip-dot --skip-tilde wget 2>/dev/null) || {
      /bin/echo "[*] Try installing wget and check PATH var"
      /bin/echo "[*] Exiting.."
      exit 1
}

# GPG/MD5 check, this can be done on yum.conf on clients too
# If enabled, bad packages will be renamed to with .BAD extentsion
# before repository structure creation/update
# This is done using "rpm -K package" command
# 0 = disable 1=enable
GPGCHECK=1

# Createrepo support for FC3+ compaibility
YUMARCH=$(/usr/bin/which --skip-dot --skip-tilde yum-arch 2>/dev/null)

CREATEREPO=$(/usr/bin/which --skip-dot --skip-tilde createrepo
2>/dev/null)
```

```
[ -z "$YUMARCH" -a -z "$CREATEREPO" ] && {
      /bin/echo "[*] Try installing yum-arch or createrepo programs and
check PATH var"
      /bin/echo "[*] Exiting.."
      exit 1
}
[ 1$(/usr/bin/id -u) -eq 10 ] && {
      /bin/echo "[*] Why running this as superuser? Try as a normal
user and check"
      /bin/echo "[*] write permissions to local repository
directories."
      /bin/echo "[*] Exiting.."
      exit 1
}
RPM=$(/usr/bin/which --skip-dot --skip-tilde rpm 2>/dev/null) || {
      /bin/echo "[*] Try installing RPM and check PATH var"
      /bin/echo "[*] Exiting.."
      exit 1
}
# Check presence of public GPG key(s)
$RPM --quiet -q gpg-pubkey || {
      /bin/echo "[*] No public GPG key(s) installed."
      /bin/echo "[*] Try to execute: rpm --import /
usr/share/rhn/GPG_KEY_FILE"
      exit 1
}
PID_FILE=/tmp/.yum_repository.pid
# Check if already running
[ -s "$PID_FILE" ] && {
      /bin/echo "[*] PID File exists $PID_FILE"
      /bin/echo "[*] Checking PID.."
      PID=$(/bin/egrep -o "^[0-9]{1,}" $PID_FILE)
      /bin/ps --pid $PID && {
            /bin/echo "[*] Process $PID found."
            /bin/echo "[*] Script seems to be already running!"
            /bin/echo "[*] Exiting.."
            exit 1
      }
      /bin/echo "[*] Process ID $PID not found"
      /bin/echo "[*] Starting new process.."
}
```

```
# Check proxy options
[ -n "$PROXY_SERVER" -a -n "$PROXY_PORT" ] && {
      # Exporting for wget
      export http_proxy="$PROXY_SERVER:$PROXY_PORT" && PROXY_FLAG=1
      WGET_ARGS="$WGET_ARGS --proxy=on"
      [ -n "$PROXY_USER" -a -n "$PROXY_PASS" ] && WGET_ARGS="$WGET_ARGS
\
      --proxy-user=$PROXY_USER --proxy-passwd=$PROXY_PASS"
}
# No process running, starting new one
/bin/echo $$ > $PID_FILE
# Sets umask
umask $DEF_UMASK
# Starts from 1st mirror definition
count=0
while [ ${MIRROR_URL[count]} ]; do
      # Some checking
      [ -d ${MIRROR_DIR[count]} ] || {
            /bin/echo "[*] Try creating localdir ${MIRROR_DIR[count]}"
            /bin/echo "[*] Exiting.."
            exit 1
      }
      [ -w ${MIRROR_DIR[count]} ] || {
            /bin/echo "[*] Check write permissions on localdir
${MIRROR_DIR[count]}"
            /bin/echo "[*] Exiting.."
            exit 1
      }
      cd ${MIRROR_DIR[count]}
      CUT_DIRS=$(/bin/echo "${MIRROR_URL[count]}" | /bin/egrep -o "\/"
| /usr/bin/wc -l)
      CUT_DIRS=$((CUT_DIRS-3))
      /bin/echo -e "[*] Writing logs to $LOG_FILE"
      /bin/echo -e "[*] Getting files from ${MIRROR_URL[count]}"
      /bin/echo -n "[*] Download started: " >> $LOG_FILE
      /bin/date >> $LOG_FILE
      # Capture some intersting signals
      trap "{
            /bin/echo \"[*] Removing PID file..\"
```

```
            /bin/rm -f $PID_FILE
            [ 1$PROXY_FLAG -ne 1 ] && {
                  /bin/echo \"[*] Unseting http_proxy var..\"
                  unset http_proxy
            }
            /bin/echo -e \"[*] Exiting..\"
            exit 1
      }" 2 3 15 19
      eval $WGET $WGET_ARGS --cut-dirs $CUT_DIRS ${MIRROR_URL[count]}
      /bin/echo -e "[*] Download complete for ${MIRROR_URL[count]}\n"
>> $LOG_FILE
      /bin/echo -e "[*] Download complete for ${MIRROR_URL[count]}\n"
      # md5 and gpg signature check
      # any package that fails this check will be renamed with
extension .BAD
      [ 1$GPGCHECK -eq 11 ] && {
            for rpm in `find ${MIRROR_DIR[count]} -name "*.rpm"`; do
                  $RPM -K $rpm >> $LOG_FILE || {
                        /bin/echo "[*] Bad RPM found: $rpm"
                        /bin/echo "[*] Moving to $rpm.BAD"
                        /bin/echo -e "\n[*] BAD package found: $rpm\n"
>> $LOG_FILE
                        /bin/mv -f $rpm $rpm.BAD
                  }
            done
      }
      for PROG in $YUMARCH $CREATEREPO; do
            # create repository dirs
            /bin/echo -e "[*] Executing $PROG on ${MIRROR_DIR[count]}"
            /bin/echo -n "[*] Time started: " >> $LOG_FILE
            /bin/date >> $LOG_FILE
            eval $PROG ${MIRROR_DIR[count]} >> $LOG_FILE 2>&1
      done
      /bin/echo -e "[*] Repository creation complete for ${MIRROR_DIR
[count]}\n" >> $LOG_FILE
      /bin/echo -e "[*] Repository creation complete for ${MIRROR_DIR
[count]}\n"
      /bin/echo -e "[*] Done.\n\n"
```

```
      count=$((count+1))
done
/bin/chmod 600 $LOG_FILE
/bin/rm -f $PID_FILE
[ 1$PROXY_FLAG -ne 1 ] && unset http_proxy

/bin/echo "[*] Finished"
/bin/echo -n "[*] Finished on " >> $LOG_FILE
/bin/date >> $LOG_FILE
exit 0
```

SUDO Configuration File: */etc/sudoers*

```
root  ALL=(ALL) ALL
Defaults    syslog=auth, logfile=/var/log/sudolog
u401  proteus=(ALL)     ALL
```

OpenSSH Server Configuration File: */etc/ssh/sshd_config*

```
Protocol 2
ListenAddress 192.168.1.200
SyslogFacility AUTHPRIV
PermitRootLogin no
X11Forwarding yes
AllowUsers u401@192.168.1.48
Banner /etc/issue.net
```

Apache Configuration File: */etc/httpd/conf/httpd.conf*

```
ServerTokens Prod
ServerRoot "/etc/httpd"
PidFile run/httpd.pid
Timeout 300
KeepAlive Off
MaxKeepAliveRequests 100
KeepAliveTimeout 15
<IfModule prefork.c>
StartServers       8
MinSpareServers    5
MaxSpareServers   20
MaxClients       150
MaxRequestsPerChild  4000
</IfModule>
<IfModule worker.c>
StartServers         2
```

```
MaxClients         150
MinSpareThreads     25
MaxSpareThreads     75
ThreadsPerChild     25

MaxRequestsPerChild  0
</IfModule>
Listen 80
LoadModule access_module modules/mod_access.so
LoadModule include_module modules/mod_include.so
LoadModule log_config_module modules/mod_log_config.so
LoadModule mime_magic_module modules/mod_mime_magic.so
LoadModule cern_meta_module modules/mod_cern_meta.so
LoadModule expires_module modules/mod_expires.so
LoadModule deflate_module modules/mod_deflate.so
LoadModule headers_module modules/mod_headers.so
LoadModule usertrack_module modules/mod_usertrack.so
LoadModule setenvif_module modules/mod_setenvif.so
LoadModule mime_module modules/mod_mime.so
LoadModule autoindex_module modules/mod_autoindex.so
LoadModule negotiation_module modules/mod_negotiation.so
LoadModule dir_module modules/mod_dir.so
LoadModule alias_module modules/mod_alias.so
LoadModule rewrite_module modules/mod_rewrite.so
LoadModule cache_module modules/mod_cache.so
LoadModule disk_cache_module modules/mod_disk_cache.so
LoadModule file_cache_module modules/mod_file_cache.so

LoadModule mem_cache_module modules/mod_mem_cache.so
Include conf.d/*.conf
User apache
Group apache
ServerAdmin root@localhost
ServerName proteus
UseCanonicalName Off
DocumentRoot "/var/www/html"
<Directory />
    Options -FollowSymLinks
    AllowOverride None
</Directory>
<Directory "/var/www/html">
    Options -Indexes -FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
<IfModule mod_userdir.c>
    UserDir disable
</IfModule>

DirectoryIndex index.html index.html
```

```
AccessFileName .htaccess
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
TypesConfig /etc/mime.types
DefaultType text/plain
<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>
HostnameLookups Off
ErrorLog logs/error_log
LogLevel warn
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}
i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog logs/access_log combined
ServerSignature Off
Alias /repository "/var/repository/"
<Directory "/var/repository">
      Options Indexes
      AllowOverride None
      Order deny,allow
      Allow from 192.168.1.0/24
      Deny from all
</Directory>
<IfModule mod_dav_fs.c>
    DAVLockDB /var/lib/dav/lockdb
</IfModule>
<IfModule mod_dav.c>
    LimitXMLRequestBody 131072
</IfModule>
IndexOptions SuppressIcon SuppressLastModified SuppressRules
VersionSort
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
```

```
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^
DefaultIcon /icons/unknown.gif
ReadmeName README.html
HeaderName HEADER.html
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
AddLanguage ca .ca
AddLanguage cs .cz .cs
AddLanguage da .dk
AddLanguage de .de
AddLanguage el .el
AddLanguage en .en
AddLanguage eo .eo
AddLanguage es .es
AddLanguage et .et
AddLanguage fr .fr
AddLanguage he .he
AddLanguage hr .hr
AddLanguage it .it
AddLanguage ja .ja
AddLanguage ko .ko
AddLanguage ltz .ltz
AddLanguage nl .nl
AddLanguage nn .nn
AddLanguage no .no
AddLanguage pl .po
AddLanguage pt .pt
AddLanguage pt-BR .pt-br
AddLanguage ru .ru
AddLanguage sv .sv
AddLanguage zh-CN .zh-cn
AddLanguage zh-TW .zh-tw
LanguagePriority en ca cs da de el eo es et fr he hr it ja ko ltz nl nn
no pl pt pt-BR ru sv zh-CN zh-TW
ForceLanguagePriority Prefer Fallback
AddDefaultCharset UTF-8
AddCharset ISO-8859-1  .iso8859-1  .latin1
AddCharset ISO-8859-2  .iso8859-2  .latin2 .cen
AddCharset ISO-8859-3  .iso8859-3  .latin3
AddCharset ISO-8859-4  .iso8859-4  .latin4
AddCharset ISO-8859-5  .iso8859-5  .latin5 .cyr .iso-ru
```

```
AddCharset ISO-8859-6  .iso8859-6  .latin6 .arb
AddCharset ISO-8859-7  .iso8859-7  .latin7 .grk
AddCharset ISO-8859-8  .iso8859-8  .latin8 .heb
AddCharset ISO-8859-9  .iso8859-9  .latin9 .trk
AddCharset ISO-2022-JP .iso2022-jp .jis
AddCharset ISO-2022-KR .iso2022-kr .kis
AddCharset ISO-2022-CN .iso2022-cn .cis
AddCharset Big5        .Big5       .big5
AddCharset WINDOWS-1251 .cp-1251   .win-1251
AddCharset CP866       .cp866
AddCharset KOI8-r      .koi8-r .koi8-ru
AddCharset KOI8-ru     .koi8-uk .ua
AddCharset ISO-10646-UCS-2 .ucs2
AddCharset ISO-10646-UCS-4 .ucs4
AddCharset UTF-8       .utf8
AddCharset GB2312      .gb2312 .gb
AddCharset utf-7       .utf7
AddCharset utf-8       .utf8
AddCharset big5        .big5 .b5
AddCharset EUC-TW      .euc-tw
AddCharset EUC-JP      .euc-jp
AddCharset EUC-KR      .euc-kr
AddCharset shift_jis   .sjis
AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz
AddHandler type-map var
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
Alias /error/ "/var/www/error/"
<IfModule mod_negotiation.c>
<IfModule mod_include.c>
    <Directory "/var/www/error">
        AllowOverride None
        Options IncludesNoExec
        AddOutputFilter Includes html
        AddHandler type-map var
        Order allow,deny
        Allow from all
        LanguagePriority en es de fr
        ForceLanguagePriority Prefer Fallback
    </Directory>
</IfModule>
</IfModule>
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-
1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
BrowserMatch "Microsoft Data Access Internet Publishing Provider"
```

```
redirect-carefully
BrowserMatch "^WebDrive" redirect-carefully
BrowserMatch "^WebDAVFS/1.[012]" redirect-carefully
BrowserMatch "^gnome-vfs" redirect-carefully
RewriteEngine On
RewriteCond %{REQUEST_METHOD} ^(TRACE|POST|OPTIONS)
RewriteRule .* - [F]
```

Logrotate Configuration File: */etc/logrotate.conf*

```
weekly
rotate 4
create

include /etc/logrotate.d

/var/log/wtmp {
    monthly
    create 0664 root utmp
    rotate 1
}

/var/log/sudolog {
        create 0600 root root
        compress
        nomail
        missingok
        notifempty
        rotate 5
        size 1M
}

/var/log/yum_repository.log {
        create 0664 root repoadm
        compress
        nomail
        missingok
        notifempty
        rotate 2
        size 5M
}
```

Yum Configuration File for Server: */etc/yum.conf*

```
[main]
cachedir=/var/cache/yum
debuglevel=2
logfile=/var/log/yum.log
pkgpolicy=newest
distroverpkg=redhat-release
```

```
tolerant=1
exactarch=1
retries=20

[base]
name=Fedora Core $releasever - $basearch – Base
baseurl=file:///var/repository/linux/fedora/$releasever/os/

[updates-released]
name=Fedora Core $releasever - $basearch - Released Updates
baseurl=file:///var/repository/linux/fedora/$releasever/updates/

[plus]
name=Fedora Core $releasever - $basearch – Extra Packages
baseurl=file:///var/repository/linux/fedora/$releasever/plus/
```

Yum Configuration File for Network Clients: */etc/yum.conf*

```
[main]
cachedir=/var/cache/yum
debuglevel=2
logfile=/var/log/yum.log
pkgpolicy=newest
distroverpkg=redhat-release
tolerant=1
exactarch=1
retries=20

[base]
name=Fedora Core $releasever - $basearch – Base
baseurl=http://192.168.1.200/repository/linux/fedora/$releasever/os/

[updates-released]
name=Fedora Core $releasever - $basearch - Released Updates
baseurl=http://192.168.1.200/repository/linux/fedora/$releasever/updates/

[plus]
name=Fedora Core $releasever - $basearch – Extra Packages
baseurl=http://192.168.1.200/repository/linux/fedora/$releasever/plus/
```