Installing a Secure Red Hat 7.1 Syslog Server

Practical Assignment
Version 1.8

Jack Green

Submitted December 2001
to fulfill GIAC GCUX requirements

Introduction

The purpose of this guide is to build upon the knowledge base of creating secure Red Hat servers. In addition, it will discuss the implementation of a syslog client that offers an additional measure of security first described by (Hines, 1999).

Currently there are papers at the *SANS* site on securing a Red Hat 7.1 laptop (DePriest, 2001), a DNS/Mail server (Pryor, 2001) and Audit reviews (Petersen, 2001). Additionally Koconis (2001) provides a guide to implementing a secure Red Hat 7.0 web server.

This paper offers a guide to implementing a secure Red Hat 7.1 syslog server. The server provides centralized logging services, basic log analysis, and secure access control and firewall protection of its logging resources.

Syslog Daemon

The syslog daemon routes messages from the kernel and from applications. These messages can range from critical errors where the kernel reports a device failure to informational messages such as a connection from an FTP client. Syslogd provides a method for classifying the information based on:

1) the facility issuing the message

Facility	Description
USER	user process
MAIL	Mail
DAEMON	miscellaneous system daemon
AUTH	Security (authorization)
SYSLOG	Syslog
LPR	Central printer
NEWS	Network news
UUCP	UUCP
CRON	Cron and At
AUTHPRIV	Private security (authorization)
FTP	Ftp server
LOCAL0-7	Locally defined
sys9-14	Typically used for Cisco Routers

2) the severity of the message

.

Severity	Description
EMERG	System is unusable.
ALERT	Action on the message must be taken immediately
CRIT	Critical condition
ERR	The message describes an error
WARNING	A warning.
NOTICE	A normal but important event
INFO	Informational
DEBUG	Debugging message.

Additionally, once categorized, differential action may be taken. These action include:

1) Write to the system console
2) Mail to a specific user
3) Write to a log file
4) Pass to another daemon
5) Discard

Consider the following partial syslog.conf file

```
# Log all kernel messages to the system console.
# Note that facility and severity are separated by a dot (.)
# Tabs separate facility.severity from action
kern.*                  /dev/console

# Log anything (except mail) of level info or higher (*.info)
# Don't log private authentication messages
# facilities may be separated by a semi-colon
*.info;mail.none;authpriv.none;cron.none        /var/log/messages

# The authpriv messages go to root.
authpriv.*                                      root

# Log all the mail messages except info (!=) in one maillog.
mail.!=info                                             /var/log/maillog

# Everybody gets emergency messages, plus log them on another
# machine parsed though a comma.
*.emerg                                                 *,@someremoteloghost

# Log cron stuff exclusively at the alert level (=) to a remote loghost
cron.=alert                                             @someremoteloghost
```

Man page syslogd(8) contains this information on your local machine. The last line shows cron alerts going to the remote syslog server. The remote server would then be responsible for storing log messages from multiple hosts (including itself) into various devices.

Network Topography

For the purposes of this discussion a host that is configured to send log files to our syslog server is referred to as a client. The server is our syslog server.

The test lab resides on a private network connected by dumb 10 mb hubs has the following hosts, roles and addresses.

Host	Role	Address
roac	General logging client	192.168.3.51
bilbo	Syslog Server	192.168.3.50
Watcher	Firewall	192.168.3.254

Roac's role could be that of any friendly server or workstation residing on our network. The purpose of this unit is to demonstrate a method for adding security to the logging process. Roac is currently running Red Hat 7.1 with a *modified* sysklogd-1.4 daemon for logging system and kernel messages.

Bilbo will serve as the hardened secure log server. It too, is running Red Hat 7.1.

Hardware list

Roac

- Dell Pentium 4 1.3 ghz
- ATA disk controller 9.gb drives
- 128 mb RAM
- 3c905 Ethernet

Bilbo

- Dell 5100 Pentium 150
- Adaptec 1542C Scsi disk controller
- 64 MB RAM
- two 4 gb Disk
- 3c509 Ethernet

Watcher

- Firewall (confidential)

Risk Analysis

A centralized log server provides several benefits

- A single location from which the system administrator may view and analyze logs for a number of machines.
- When an individual machine comes under attack, among the first things the *bad guy* may do is to conceal any of his/her logged activity. The centralized logger can provide a log of the attack and subsequent measures taken by the attacker independent of the compromised machine.
- Logs can consume great gouts <g> of disk space on local drives. Transferring logging will decrease the probability of filling up client disk space. Additionally the administrator is responsible for closely monitoring the free space of only one host.
- A secured log can carry greater evidentiary weight as it resides in a controlled environment.

The key threats of running a log server include

- All logged information is on one server. Once that host is compromised, its information becomes available to the attacker.
- There are DOS attacks that can fill the log partition,
- Should the log server go down, all logging is down.
- Since log messages are sent udp/514 over the net, a sniffer can easily intercept log messages.
- Since log messages are sent udp, messages are transmitted over an unreliable connection.

Most of these threats can be mitigated with the measures described in this paper.

- We will take steps to minimize the risk of the log host being compromised.
- We will take steps to minimize the risk of a DOS attack
- One may purchase fault-tolerant hardware
- Using syslog-ng messages are transmitted using tcp (among other features). This paper will not address the use of syslog-ng (nsyslogd is not ready for Linux yet). It will explore the use of sysklogd on the client-side and a hardened implementation of Linux on the server-side.

Installing a secure syslog client – step by step

Since this technique will work with any client, we won't go through a set-by-step installation of the Red Hat machine. It is given that the machine is up and performing its functions on the network.

This technique, described by Eric Hines[1] , involves a clever ruse where syslogd reads its settings from a masqueraded syslog.conf file. The source code for the syslogd file is reconfigured to read another, renamed conf file. Should an intruder gain control over the system, they see the syslog.conf file in place but not logging anything. In fact, the logging is disguised but going elsewhere[2].

1) Retrieve the source code.

ftp.redhat.com/redhat/linux/7.1/en/os/i386/RedHat/RPMS/sysklogd-1.4-4.src.rpm

2) Present SOURCES

```
rpm –i –iv sysklogd-1.4-4.rpm
```

The sources code will be moved into /usr/src/redhat/SOURCES/

3) Unzip/untar the file

```
tar xvf sysklogd-1.4rh.tar.gz
```

4) Use your favorite editor to open syslog.c

```
vi syslog.c
```

5) Find _PATH_LOGCONF change syslog.conf to another name and save it

```
grep (/) _PATH_LOGCONF
R(eplace) /etc/syslog.conf with   /etc/.sys/CORE.conf
esc :wq
```

6) tar up the directory again

```
tar cvf sysklogd-1.4rh.tar.gz ./sysklogd-1.4rh
```

[1] Hines, Eric. Complete Reference Guide to Creating a Remote Log Server. http://www.linuxsecurity.com/feature_stories/remote_logserver-1.html. 22 AUG, 2000

[2] An editor correctly pointed out that an attacker may run a *strings* against the binary and find a logging path. The author offers this section as an additional countermeasure as well as an exercise in pure professional curiosity.

7) rpm the file to build the binaries syslogd and klogd

```
rpm –tc sysklogd-1.4rh.tar.gz
rpm –bc sysklogd-1.4rh.tar.gz
rpm –bi sysklogd-1.4rh.tar.gz
rpm –bb sysklogd-1.4rh.tar.gz
```

8) Rename the current syslogd and klogd that reside in /sbin

```
cd /sbin
mv syslogd syslogd.old
mv klogd klogd.old
```

9) Create the directory and file for CORE.conf

```
mkdir /etc/.sys
cp /etc/syslog.conf /etc/.sys/CORE.conf
```

10) add bilbo (loghost) to etc/hosts file

```
cd /etc
vi hosts
I 192.168.3.50              bilbo
esc : wq
```

11) Edit CORE.conf to write to Bilbo.[3]

```
# To forward all kernel messages to Bilbo the configuration file would be
as follows:

 kern.*         @Bilbo
```

12) Make Core.conf *rw* by root only.

```
chmod +600 CORE.conf
```

13) Complete our ruse by commenting out all entries in the syslog.conf file

```
# To comment out logging the kernel messages configuration file would
have a # in front of it:
 # kern.*         /dev/console
        .
```

[3] For more information on the configuring the CORE.conf file, refer to the section **Syslog Daemon** or to Appendix C for a sample.

It should be noted that an alternative is to log locally and to our log server. In that case, there are two sets of logs which may be compared in the event of an intrusion. To do so, don't comment out the logging but add a second line, delimited by a comma. For example:

```
# To forward all kernel messages to Bilbo and locally log to the file
kernel, the configuration file would be as follows:
      kern.*                         @Bilbo, /var/adm/kernel
```

14) Copy the revised binaries to /sbin

```
cd /usr/src/redhat/BUILDS/sysklogd-1.4rh
cp klogd /sbin/klogd
cp syslogd /sbin/syslogd
```

15) Lets change the *apparent* creation date to match the others in /sbin

```
touch -m 02072001 /sbin/syslogd
```

16) Restart the syslog daemon

```
/etc/rc.d/init.d/syslog restart
```

17) Test our facility

Initlog –s "test" (or)
Logger "test"

The client is now ready to log to our log server, Bilbo

Log Server Configuration

Preparation

- Have an envelop ready in which you will store the root password.
- If your organization doesn't currently have ntp time servers, we need to get permission to use public facilities.

1) Go to the public time server site and select three secondary servers: http://www.eeics.udel.edu/~mills/ntp/servers.htm

2) Get permission to connect from the listed administrator for each of these three sites.

3) List sites here as permission is received

	Permission date
Site 1	
Site 2	
Site 3	

Install Red Hat

Prerequisites:

- Ensure server network card(s) are disconnected.
- Gather information required below

System need	User supplied value
Hostname	
IP address	
Subnet mask	
Gateway	
DNS	

Insert Red Hat CD-1 as system powers up. It should boot to CD. If not, see BIOS setup section. Supply the appropriate values

- At LILO: Type **expert**
- Supply the driver disk if you have a specific need, e.g., custom RAID
- Choose your language
- Choose Mouse
- Installation type should be **Custom**
- Manual Partition with druid – As a log server, your system should be heavy on disk, giving as much space to /var as possible.
-

Mount point	**Size**	**Comments**
/	1000MB	Will be sufficient for root partition
/home	1000MB	We are supporting only two users
<Swap>	256MB	2 * RAM
/usr	500MB	
/tmp	500MB	
/var	Remainder	Ideally, this is a separate spindle to prevent shutdown in the event of an unthwarted DOS attack.

- Format all partition and check for bad blocks
- LILO (LInux LOader)
  - Install on MBR – there should be no other OS's on you log server
  - Create a boot diskette
  - Keep remaining defaults
- Network Configuration
  - Select activate on boot
  - Uncheck DHCP
  - Supply network information
- Firewall configuration
  - Choose high
  - Choose customize – allow SSH,123:udp(ntp),514:udp (Syslog) – comma separate each port as shown
- Language support
  - You may choose to support more than you're preferred language
- Time setting
  - Choose Coordinated Universal Time (UTC).
  - Select the time zone
- Account Configuration
  - Choose a strong password[4], write it down and put it in the envelop, seal it, label it and put it in a safe place
  - Add an admin user
- Authentication
  - Enable md5
  - Enable shadow
  - Do not enable NIS
  - Do not enable LDAP
  - Do not enable Kerberos
- Select individual package installation

Amusements		
	Games	None

[4] The SANS GSEC material suggests a password with at least one alpha, one numeric and one special character. Additionally, the root password may be further obfuscated by taking the first letter of each word from an easily remembered seven or eight word phrase, e.g., Pick a number between 1 and 9 = Panb1&9.

	Graphics	None
Applications		
	Archiving	dump-static
		pax
		rmt
	Communications	None
	Databases	None
	Editors	Add vim
	Engineering	None
	File	Add stat
	Internet	ftp
		openssh
		tcpdump
		traceroute
		wget
	Multimedia	None
	Productivity	None
	Publishing	None
	System	sysstat
		tripwire
		vlock
	Text	None
Development		
	Debuggers	Add lsof
	Languages	python
	Libraries	python-xmlrpc
		rpm-python
	System	None
	Tools	make
Documentation		Add man pages
System Environment	Base	iptables
		rhn_register
	Daemons	iputils
		ntp
		openssh-server
		tcp-wrappers
		xinetd
	Kernel	None
	Libraries	gmp
	Shell	None
User interface		
	Desktops	None
	X	None
	X Hardware support	None

Format and feed CD – 2 to it when prompted.
Reboot

Ensure that small services are disabled in /etc/xinetd.d.

Small services, especially their UDP versions, are unlikely to be used, but can be used to launch denial of service and other attacks. They should be disabled. The following files are included in xinetd.d. Ensure the line *disable* = *yes* is contained in these files

	chargen
	chargen-udp
	daytime
	daytime-udp
	echo
	echo-udp
	time
	time-udp

Configure our log server to listen

By default the Linux syslog daemon logs only locally. We must set syslogd to receive (-r). Refer to Appendix A for an example of a typical syslog script.

1) Open the syslog script

```
cd  /etc/rc.d/init.d/
vi syslog
```

2) Find the line SYSLOG_OPTIONS and insert the argument *–r*

```
/ SYSLOGD_OPTIONS
I –r
esc :wq
```

Configure the logserver's syslog.conf

Referring to appendix B,the default syslog.conf file provided by Red Hat, there are a couple of additions worthy of comment.

Line 1 (excludes comments) shows all warnings and errors are being logged to /var/log/syslog

```
kern.*                                  /var/log/kernel
```

Line 8 shows all kernel messages are being logged to /var/log/kernel.

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

Since these are not default lines, create and set security on these files.

```
touch /var/log/kernel /var/log/syslog
chmod 700 /var/log/syslog  /var/log/kernel
```

Configure our log server to rotate logs

1) Your organization should have a policy regarding how long logs must be retained. Normal rotation is set for sixty days. Let’s change it to a year.

```
cd /etc
vi logrotate.conf
```

2) Find the line below *rotate log files* and change to *monthly* (if not already set)

```
# rotate log files …
  monthly
```

3) find the line that says *rotate 4* and change to *rotate 12*. You should also change the comments. Programmers *always* document, right?

```
# keep 4 weeks worth of backups
   rotate 12
```

Turn off all INETD Services

> The **inetd** program listens for connections on certain internet sockets. When a connection is found on one of its sockets, it decides what service the socket corresponds to, and invokes a program to service the request. The server program is invoked with the service socket as its standard input, output and error descriptors.

Essentially, **inetd** allows running one daemon to invoke several others, reducing load on the system. Inetd is typically started at runlevel 3,4,5.

tool to simplify management of the symbolic links in /etc/rc.d under Linux.

1) Since we don't need inet services, let's shut them off.

```
chkconfig inet off
```

Installing ssh

"Ssh is a program for logging into a remote machine and for executing commands on a remote machine. It is intended to replace rlogin and rsh, and provide secure encrypted communications between two untrusted hosts over a secure network."[5]

Sniffers are easy to use and telnet provides clear transmission. The *r* utilities are known for their dangers. Our administrator will need a secure shell to gain access to the log server so we'll install ssh.

Recall the we installed the RPM's for openssh server and client during setup. Now lets configure our server for secure remote access for the administrator. Should choose to use the version from openssh.com, appendix D provides an example startup script.

The OpenSSH daemon uses the configuration file /etc/ssh/sshd_config. The default configuration file installed is suitable with one exception.

1) There is no reason to allow another (mailicious) user on our local network the opportunity to log on as root. Furthermore anything that an administrator does should be run under his/her name using *su* to obtain the necessary root permission. Deny root login by changing PermitRootLogin from *yes* to *no*

```
cd /etc/ssh
 vi sshd_config
 PermitRootLogin no
 esc:wq
```

2) Ensure the ssh daemon is set for startup and start/restart it

```
chkconfig sshd on
/etc/rc.d/init.d/sshd restart
```

[5] SSH(1) Unix man page.

3) Setup hosts.allow and hosts.deny. Use your editor to allow admin secure shell access to the loghost from our internal net.

```
cd /etc

vi hosts.allow
(A)ppend sshd: LOCAL
esc:wq

vi hosts.deny
(A)ppend ALL: ALL
esc:wq
```

Configure ntp

You've received permission from the timemasters for your secondary timeservers.

1) If you're not running DNS add the timeservers into your /etc/hosts file.

```
vi /etc/hosts

xxx.xxx.xxx.xxx     selection1
yyy.yyy.yyy.yyy     selection2
zzz.zzz.zzz.zzz     selection3
```

2) Set up /etc/ntp.conf. The first two lines are for the system clock. Open **vi /etc/ntp.conf** and enter the lines in this format where selection(n) is the FQDN for the site

```
server      127.127.1.0
# local machine's clock
fudge       127.127.1.0 stratum 10
server      selection1 prefer    #timekeeper email
server      selection2           #timekeeper email
server      selection3           #timekeeper email
driftfile /etc/ntp.drift
```

3) Save and exit vi

```
esc:wq
```

4) Tighten perms, if necessary

```
chmod 640 /etc/ntp.conf
```

5) Create the drift file

```
touch /etc/ntp.drift
```

6) Tighten perms, if necessary

```
chmod 640 /etc/ntp.drift
```

7) Start ntpd

```
/etc/rc.d/init.d/ntpd start
```

8) Set cron to update the system clock where selection1 is the preferred secondary timeserver's FQDN and start it

```
/usr/sbin/ntpdate –b selection1
/sbin/chkconfig ntpd on
```

Prepare to bring the system onto the network

1) Discover which daemons are listening on TCP

```
netstat –at
```

```
Active Internet connections (servers and established)

Proto Recv-Q Send-Q Local Address     Foreign Address        State
tcp        0      0 *:ssh             *:*           LISTEN
tcp        0      0 bilbo.smtp        *.*           LISTEN
```

2) Shut off unnecessary sendmail service

```
/etc/rc.d/init.d/sendmail stop
chkconfig sendmail off
```

3) Check TCP again

```
netstat –at
```

```
Active Internet connections (servers and established)

  Proto Recv-Q Send-Q Local Address           Foreign Address         State
  tcp        0      0 *:ssh                   *:*                     LISTEN
```

4) Discover which daemons are listening on UDP

```
netstat –au
```

```
Active Internet connections (servers and established)

Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp       0     0       *:syslog                      *:*
udp       0     0       192.168.3.51:ntp              *:*
udp       0     0       bilbo:ntp                     *:*
udp       0     0       *:ntp                         *:*
```

5) Stop ssh for the time being

```
/etc/rc.d/init.d/sshd stop
```

6) Stop syslog

```
/etc/rc.d/init.d/syslog stop
```

7) Stop ntp

```
/etc/rc.d/init.d/ntpd stop
```

8) Discover which daemons are *still* listening

```
netstat –at
```

```
Active Internet connections (servers and established)

Proto Recv-Q Send-Q Local Address       Foreign Address         State
```

9) Discover which daemons are listening on UDP

```
netstat –au
```

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
```

Install updates

Once no internet connections are listening, we should bring the log server on line and get our updates. Since we set the eth0 interface to start at boot, we need only plug it in and watch for a link light.

Connect to the network by plugging in the Ethernet cable.

Download the appropriate RPM packages

1) Create directories for the updates. It is a good practice to keep updates together.

```
mkdir –m 700 /usr/local/updates
mkdir –m 700 /usr/local/updates/kernel
```

2) Change to the usr/local/updates directory

```
cd  /usr/local/updates
```

3) Create a sorted file (or a printout) of the installed packages

```
rpm –qa |sort>srtlisting
```

4) We'll build upon a good concept suggested by Pryor (2001) and open a second tty. Using *more,* we'll list our alpha sort of installed packages and use it as a reference against the rpm's available at redhat.com. If we had the bandwidth, we could download all updates and, as you'll see, perform a *freshen* without using this step.

```
Alt-f2
login
cd /usr/local/updates
more srtlisting
```

From tty1 Red Hat will offer a package. From tty2 we'll look at the list and choose whether or not we need the package.

5) Alt-F1 back to the ftp session, ftp to redhat.com and issue the mget command

```
[/usr/local/updates]# ftp
ftp> open updates.redhat.com
Log in as anonymous with your email address as your password.
ftp> cd 7.1/en/os/i386
  mget *
```

Choose (y)es or (n)o to download update RPM's move between the two ttys' to select those which are relevant to you particular install.

Apply the updates

Move the kernel updates into the /usr/local/updates/kernel since they will be installed, not updated.

```
mv kernel-* kernel/
```

Apply the updates with the *freshen* argument. Freshen will apply only those updates that are relevant.

```
rpm –F /usr/loca/updates/*
```

Apply Kernel updates

The kernel updates should be applied with the –ivh option rather than the U option. The U option will remove the previous version of the kernel. We should retain the original should the updated kernel not behave well.

1) Run rpm on each file in the kernel directory. As of Early December 2001:

```
rpm –ivh kernel-2.4.9-12.i386.rpm
rpm –ivh kernel-BOOT-2.4.9-12.i386.rpm
rpm –ivh kernel-headers-2.4.9-12.i386.rpm
rpm –ivh kernel-source-2.4.9-12.i386.rpm
rpm –ivh kernel-doc-2.2.9-12.i386.rpm
```

2) Point lilo to both kernels. Edit /etc/lilo.conf

```
image=/boot/vmlinuz
        label=linux
        read-only
        root=/dev/sda1
```

```
                image=/boot/vmlinuz-2.4.2-2
                        label=linux
                        read-only
                        root=/dev/sda1
```

3) Run LILO and reboot

```
        /sbin/lilo
        /sbin/reboot
```

Register the system to notify for updated errata

Our log server won't be available for AutoPRM's once its locked down. However we can make os maintenance a great deal easier by registering our system with redhat.com. As an *entitled* system, we can receive notifications of updates via email.

Run rhn_register as root

```
        /usr/sbin/rhn_register
```

You will be guided through the process of creating an account (or logging in). After which the Red Hat will profile your installation. You may choose to ignore certain RPM updates. For example, you'll not want to freshen your install of the modified client sysklogd package. That update will need to be done manually.

Once complete, you will be notified via email regarding available updates.

Retrieve the *newperms* script for later use

While we have wget hot, let's download a script that we'll discus later

```
        wget www.sans.org/linux/newperms
```

Configure Swatch

SWATCH, "The Simple WATCHer and filter", is a perl program developed by Todd Atkins. It monitors your logs in real time and performs an designated action once a given condition is met. Installing swatch is fairly straightforward. You may find you need these rpm's to support swatch:

perl-TimeDate-1.10-1.i386.rpm

perl-TimeHiRes-01.20-9.i386.rpm
perl-FileTail-0.96-1.0.2.i386.rpm
perl-Date-Calc-4.3-1.0.2.i386.rpm

They are available in the power tools cd and, of course, on the web.

The default configuration file is .swatchrc residing in the user's home directory.

To create a swatch file, go to *your* home directory

```
cd ~
```

The general format for .swatchrc is[6]

- Keyword *watchfor* or *ignore*
- Condition, expressed as a regular expression
- Action to be performed
- Time interval to ignore the matched pattern before performing the action (optional)
- The last column (required if you are using the third field) is a timestamp, defined as start:length. This defines the location and length of the timestamp in the notification message.

We're just going to trap a few events. They are outlined in Appendix E.
Using vi create a swatch config file using the lines described in that appendix.

```
vi .swatchrc
```

To run swatch, We'll just use the command line. The swatch manual gives detail on other options. I'll log in as root and run swatch against the file (-f) /var/log/messages and append the output to the file *careful*

```
Swatch –f /var/log/messages>>careful
```

Since I've blocked the SMTP port, I can't mail output to me real-time. Hence we're doomed to using SSH or being at the console to run swatch.

Configure ipchains

[6] For a complete discussion of swatch, visit http://www.engr.ucsb.edu/~eta/swatch/swatch.html

Recalling from installation time, we specified a high security firewall ruleset. We added permitting ssh (port 22), ntp and syslog. We have a good start from installation. We must inspect our rules:

```
Ipchains --list
```

```
Chain input (policy ACCEPT):
target      prot  opt     source          destination       ports
ACCEPT      udp  ------   anywhere         anywhere          any ->  syslog
ACCEPT      tcp  -y----   anywhere         anywhere          any ->  ntp
ACCEPT      udp  ------   anywhere         anywhere          any ->  ntp
ACCEPT      tcp  -y----   anywhere         anywhere          any ->  ssh
ACCEPT      all   ------  anywhere         anywhere          n/a
ACCEPT      udp  ------   192.168.3.254   anywhere          domain ->  any
ACCEPT      udp  ------   192.168.3.5      anywhere          domain ->  any
REJECT      tcp  -y----   anywhere         anywhere          any ->  any
REJECT      udp  ------   anywhere         anywhere          any ->  any
Chain forward (policy ACCEPT):
Chain output (policy ACCEPT):
```

The two rules of interest are in bold.
The first rule says anyone can write to our syslog. Since we wish to control who logs to the log server, we will replace the 1st rule:

```
ipchains –R input 1 –s 192.168.3.50 –d 192.168.3.51 514 –p udp –j ACCEPT
```

The fourth rule would let anyone ssh into out log server. We will restrict it to the admin’s workstation.

```
ipchains –R input  4 –s 192.168.3.200 –d 192.168.3.51 22  –p tcp –j ACCEPT
```

Reviewing our rules we see they seem ready to test.

```
ipchains –list
```

```
Chain input (policy ACCEPT):
target      prot opt     source          destination       ports
ACCEPT     udp  ------  192.168.3.50    192.168.3.51      any ->  syslog
ACCEPT     tcp  -y----   anywhere        anywhere          any ->  ntp
ACCEPT     udp  ------   anywhere        anywhere          any ->  ntp
ACCEPT     tcp  -y----  192.168.3.200   192.168.3.51      any ->  ssh
ACCEPT     all  ------   anywhere        anywhere          n/a
ACCEPT     udp  ------  192.168.3.254   anywhere          domain ->  any
ACCEPT     udp  ------  192.168.3.5      anywhere          domain ->  any
REJECT    tcp  -y----   anywhere        anywhere          any ->  any
REJECT    udp  ------  anywhere      anywhere            any ->  any
Chain forward (policy ACCEPT):
Chain output (policy ACCEPT):
```

Lock down the system

Fix /etc/inittab

Disable “Control-Alt-Delete” rebooting

change:

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

to:

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

Require root password when booting to single user mode

add:

```
        ~~:S:wait:/sbin/login
```

after the entry for “si::sysinit:/etc/rc.d/rc.sysinit.”

Make the changes effective

```
        init q
```

Password protect LILO command-line options

Edit /etc/lilo.conf to include the following after the prompt entry:

```
        password = Strong-password
        restricted
```

Replace “***strong-password***” with your own password.

Change the permissions on lilo.conf

The password is stored in clear text so tighten the permission to root

```
chmod 600 /etc/lilo.conf
```

Effect the changes to lilo

```
/sbin/lilo
```

Install Warning banners

It is critical to have a login and a message of the day banner if you wish to prosecute intruders/misuers of your systems. The exact text of such messages is left to lawyers to craft. A sample used by the US Department of Energy may be found at http://www.ciac.org/ciac/bulletins/j-043.shtml. Use the banner that your legal staff wrote. It should be placed in:

```
/etc/motd
/etc/issue
/etc/issue.net
```

Ensure that no unnecessary services are running

We have been careful at installation time to install the minimum of services. To ensure you may wish to run these commands. We did so earlier and a recheck showed only sshd and syslog listening.

```
netstat –at
netstat –au
lsof -i +M
```

Tighten file permissions

A great resource available from www.sans.org, "Securing Linux Step-By-Step", includes a script that removes permissions for *regular users*. The script, written by David A. Ranch, is also available at http://www.sans.org//linux/newperms . Appendix F shows a copy of the script. Basically newperms removes read-write-execute from the others group for an array of executables in the directories /bin, /sbin, /usr/sbin. It also provide r-w permissions for lsof and lp*'s and the SUID bit for lpr. Since the script is generic to installs we won't be surprised if we see some "No such file or directory" error messages.

This script should be run after RPM updates as well so keeping it among the updates is a handy reminder.

```
cd \usr\local\updates
sh newperms
```

Remove unnecessary login accounts

Several unnecessary login accounts installed by default. lets remove these accounts.As usual, you may need some of these depending on your particular configuration so use due care.

```
mail
news
uucp
games
gopher
ftp
mailnull
nobody
xfs
```

Remove ftp

We have no further need for ftp and wget. They should be removed.

```
rpm -qa |grep ftp
        ftp-x.x.x
rpm -e ftp-x.x.x

rpm –qa |grep wget
        wget-x.x.x
rpm -e wget-x.x.x
```

Lock Down the BIOS

During POST all systems will allow you access to the BIOS. Dell systems tend to use F2. Of course, BIOS setups vary. Some will provide greater control over *pre*-boot sequences than others. If you can set these, do so. If you give a bad guy physical access to the computer, it will be owned. However, we'll slow down someone who gains physical access to our unit by:

Remove floppy and CD-ROM from the boot sequence
Ensure that case intrusion detection is turned on (might as well)
Enable password protection of Set up
Disable PXE (network boot/install)

Secure Server Room

The room in which the server resides should be locked and secured from illegal entry. There should be uninterruptible power available.

Ongoing Maintenance

O/S updates

Updated RPM's will be an ongoing task. Since we arranged for email notification, the administrator may choose the interval that is appropriate. Some updates are more critical than others.

If you have the bandwidth, an alternative is to simply download all updates for a given release with wget. Of course, you've removed ftp and wget from your syslog server, so you must use another workstation, perhaps one with write access to a cd-rw device. The command for retrieving updates follows:

```
wget  'ftp://updates.redhat.com/7.1/en/os/*.rpm'
```

Place the RPM's on a CDRW and carry them to each machine needing to be refreshed. Use the same procedure to freshen the desired RPM's as described in the *Install Updates* section above.

Disaster Recovery

Should we need to rebuild them, let's copy the master boot record and document the had drive's partition layout to a floppy..

Copy MBR to floppy

```
[/root]# dd if=/dev/sda of=/mnt/floppy/MBR.sda count=1
```

Copy partition layout

```
[/root]# fdisk –l /dev/sda > /mnt/floppy/PTBL.sda.txt
```

Let's make a mirror of our known good unit. Go into single-user mode and use the dd command to copy disk to disk. Then, go back into multi-user mode and verify copy. Finally, run fsck on all partitions to verify integrity.

```
init 1
dd if=dev/sda of=/dev/sdb bs=lk
init 3
```

```
fsck –y /dev/sdb1[2,3,4]
```

Finally you may want remove the drive and put it in a safe place.

Subscribe to security lists

While we've already subscribe to Red Hat's RPM update list there are several other resources dealing with security updates including"

BUGTRAQ Announcements
http://www.securityfocus.com

CERT Advisories
http://www.cert.org/advisories

Federal Bureau of Investigation
Contact your local office regarding the Infragard or
http://www.infragard.net/

RedHat bugfix announcements
https://listman.redhat.com/mailman/listinfo/redhat-watch-list

RedHat Security Advisories
https://listman.redhat.com/mailman/listinfo/redhat-watch-list

Security Alert Consensus
http://www.sans.org

Backups

The log server holds data critical to multiple machines and backups are important. It comes equipped with a scsi tape device.

Backups should be run on the entire system on one given day during the week.

```
cd /
tar –cf /dev/rst0 /
```

Backups of the var/log can be run on the other days.

```
cd /var/log
tar –cf /dev/rst0 /var/log
```

Configuration check

1) Services check:

We've locked Bilbo down to accepting only UDP 514 logging transactions, ntp and ssh connections. To confirm this list the open internet files::

```
lsof –i +M
```

```
COMMAND   PID    USER   FD   TYPE   DEVICE SIZE NODE NAME
syslogd   558 root    9u  IPv4    797      UDP       *:syslog
ntpd      661 root    4u  IPv4    922      UDP       *:ntp
ntpd      661 root    5u  IPv4    923      UDP       bilbo:ntp
ntpd      661 root    6u  IPv4    924      UDP       192.168.3.50:ntp
sshd      686 root    3u  IPv4    959      TCP       *:ssh (LISTEN)
```

2) ipchains check

We set ipchains to only accept syslog messages from out test workstation *roac.* Prior to setting that rule, I had my firewall (192.168.3.254) logging to Bilbo as well.

I'll test two ways:

a. The presence of a log from roac using logger. Enter the command on roac:

```
logger "test from roac"
```

Log output

```
Dec  1 10:50:34 192.168.3.254 id=firewall sn=00301E051156 time="2001-12-01
13:02:44" fw=11.110.178.118 pri=6
c=1011 m=97 n=6006 src=192.168.3.21:1639:LAN dst=216.32.120.136:80:WAN
proto=http op=GET rcvd=36410 result=200
dstname=cgi.ebay.com arg=/aw-
cgi/eBayISAPI.dll?ViewItem&item=1670565488^M

Dec  1 10:50:37 192.168.3.254 id=firewall sn=00301E051156 time="2001-12-01
13:02:47" fw=11.110.178.118 pri=6
```

```
c=1011 m=98 n=18941 src=192.168.3.21:1626:LAN dst=11.110.178.9:53:WAN
proto=udp/dns rcvd=352 ^M

Dec  1 10:50:37 192.168.3.254 id=firewall sn=00301E051156 time="2001-12-01
13:02:47" fw=11.110.178.118 pri=6
c=1011 m=97 n=6007 src=192.168.3.21:1644:LAN dst=216.119.11.120:80:WAN
proto=http op=GET rcvd=1770 result=200
dstname=abacus.sj.ipixmedia.com
arg=/abc/M28/65ca3ca5ea74e0760d3ae68756/i-2_B_T.JPG^M
```

Dec 1 10:58:09 roac root: test from roac

The first 3 line groups show logging from web activity. On closer inspection we see that my clever wife shops on line <grin>. The last line show my test message being permitted to register. There are no subsequent lines from the firewall.

b. A closed port 514 from another blocked IP. In this case 192.168.3.200. Using nmapNT from another local station on the network

```
        nmapnt –sU 192.168.3.50
```

The output shows

```
Starting nmapNT V. 2.53 SP1 by ryan@eEye.com
eEye Digital Security ( http://www.eEye.com )
based on nmap by fyodor@insecure.org  ( www.insecure.org/nmap/ )

All 1448 scanned ports on  (192.168.3.50) are: filtered
Nmap run completed -- 1 IP address (1 host up) scanned in 295 seconds
```

3**) Client logging facility is obscured**

As can be seen from the log above, my modified syslog daemon is writing to the log server. A brief review of /var/log/ show that the messages log and others have been rotated out and are empty.

4) Swatch traps our events of interest

Referring to Appendix E, recall that we configured .swatchrc to point out login activity and sshd activity failures (among others

Call swatch (the default file is .swatchrc)

```
        swatch –f /var/log/messages
```

echos to the screen (abbreviated output)

```
*** swatch-3.0.4 (pid:9371) started at Sun Dec 30 13:37:55 EST 2001

Dec 30 00:38:45 bilbo rc: Starting sshd:  succeeded
Dec 30 16:25:12 bilbo sshd: sshd -TERM succeeded
Dec 30 16:25:12 bilbo sshd:  succeeded
Dec 30 16:46:30 roac PAM_pwdb[492]: (su) session closed for user root
Dec 30 16:46:49 roac PAM_pwdb[2485]: (su) session opened for user root
by root(uid=0)
Dec 30 16:46:57 roac PAM_pwdb[2485]: (su) session closed for user root
Dec 30 16:47:10 roac PAM_pwdb[2488]: check pass; user unknown
Dec 30 16:47:19 roac PAM_pwdb[2488]: check pass; user unknown
Dec 30 16:47:29 roac PAM_pwdb[2488]: check pass; user unknown
Dec 30 16:48:22 roac PAM_pwdb[2492]: password for (joeuser/501)
changed by (root/0)
```

5) Are small-services shut down?

We disabled the small services. Doing a quick ports scan of those tcp ports less than 20 against Bilbo we see:

```
nmap –sT bilbo –p 1-20

Starting nmap V. 2.12 by Fyodor (fyodor@dhp.com, www.insecure.org/nmap)
Interesting ports on Bilbo.my.com (192.168.3.50)
(not showing ports in state:filtered)

Port    State  Protocol       Service


Nmap run completed -- 1 IP address (1 host up) scanned in 1 second
```

6) Are we displaying the banners

At login prompt and authentication time our banners must (do) display

7) Log Server does not boot from floppy or CD

After placing either the distribution CD or the Boot floppy in their respective slots, the system will only boot from the fixed disk.

Discussion

This paper presents a system for implementing a hardened Red Hat 7.1 syslog server. During the risk analysis, I discussed threats against a centralized log server. We have minimized the following threats

Threat	Countermeasure
Once a centralized log servier is compromised, the logs of all clients are available	Harden log server
Susceptible to DOS attacks	Harden log server, accept udp/514 from only selected hosts
Logs being sniffed	This problem can be minimized with tools that identify nic's set in promiscuous mode, with switches and vlans and subnetting.
Log server goes down, all logging fails.	Countermeasures included securing the server room, providing uninterruptible power and hardening the system.

Since log messages are still being published udp, clear text and unencrypted logs, further research implementing ssyslog or msyslog would be useful.

Appendix A

```
#!/bin/bash
#
# syslog        Starts syslogd/klogd.
#
#
# chkconfig: 2345 12 88
# description: Syslog is the facility by which many daemons use to log \
# messages to various system log files.  It is a good idea to always \
# run syslog.

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

[ -f /sbin/syslogd ] || exit 0
[ -f /sbin/klogd ] || exit 0

# Source config
if [ -f /etc/sysconfig/syslog ] ; then
        . /etc/sysconfig/syslog
else
        SYSLOGD_OPTIONS="-r -m 0"
        KLOGD_OPTIONS="-2"
fi

RETVAL=0

umask 077

start() {
        echo -n $"Starting system logger: "
        daemon syslogd $SYSLOGD_OPTIONS
        RETVAL=$?
        echo
        echo -n $"Starting kernel logger: "
        daemon klogd $KLOGD_OPTIONS
        echo
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/syslog
        return $RETVAL
}
stop() {
        echo -n $"Shutting down kernel logger: "
        killproc klogd
        echo
        echo -n $"Shutting down system logger: "
        killproc syslogd
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/syslog
        return $RETVAL
}
rhstatus() {
        status syslogd
        status klogd
}
restart() {
        stop
        start
}

case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  status)
        rhstatus
        ;;
  restart|reload)
        restart
        ;;
  condrestart)
        [ -f /var/lock/subsys/syslog ] && restart || :
        ;;
  *)
        echo $"Usage: $0 {start|stop|status|restart|condrestart}"
        exit 1
esac

exit $?
```

Appendix B

```
# Log server /etc/syslog.conf
#
# Log all kernel messages to /var/log/kernel
```

```
kern.*                                  /var/log/kernel

# Log anything level info or higher to messages.
*.info;mail.none;authpriv.none          /var/log/messages

# The authpriv file has restricted access.
authpriv.*                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                  /var/log/maillog

# Log cron stuff
cron.*                                  /var/log/cron

# Everybody gets emergency messages.
*.emerg                                 *

# Save boot messages also to boot.log
local7.*                                /var/log/boot.log

# Save warnings and errors to syslog
*.warn;*.err                            /var/log/syslog

# Configure realtime logging to VTY 7 and 8

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

Appendix C

```
# Logging client /etc/sys/CORE.conf
#
# Log all kernel messages to /var/log/kernel
kern.*                            @bilbo

# Log anything level info or higher to messages.
*.info;mail.none;authpriv.none    @bilbo


# The authpriv file has restricted access.
```

```
authpriv.*                                  @bilbo


# Log all the mail messages in one place.
mail.*                                  @bilbo


# Log cron stuff
cron.*                                  @bilbo


# Send emergency messages to the remote.
*.emerg                                 @bilbo


# Save warnings and errors to the remote syslog
*.warn;*.err                            @bilbo
```

Appendix D

```
#! /bin/sh
#
# chkconfig: - 345 50 50
# description: The ssh protocol allows secure logins.
# processname: rpc.rusersd
# Source function library.
. /etc/rc.d/init.d/functions
# Get config.
. /etc/sysconfig/network
# Check that networking is up.
if [ ${NETWORKING} = "no" ]
then
        exit 0
fi
# See how we were called.
case "$1" in
  start)
        echo -n "Starting ssh services: "
        daemon /usr/local/sbin/sshd

        echo
        touch /var/lock/subsys/sshd
        ;;
  stop)
        echo -n "Stopping ssh services: "
        killproc sshd
```

```
            echo
            rm -f /var/lock/subsys/sshd
            ;;
  status)
            status sshd
            ;;
  restart|reload)
            $0 stop
            $0 start
            ;;
  *)
            echo "Usage: sshd {start|stop|status|restart}"
            exit 1
esac

exit 0
```

Appendix D

```
#!/bin/sh

  PATH=/usr/sbin:/sbin:/bin:/usr/sbin

  LOCAL_INTERFACE="192.168.3.50/32" Bilbo’s IP address
  LOCAL_NETWORK="192.168.1.0/24"   # network/mask here
  SSH_PERMITTED="192.168.3.200/32" # only .200 allowed to ssh
  SYSPERMITTED="192.168.3.51/32" # Roac allowed to log messages


  # deny everything
  ipchains -P input DENY
  ipchains -P output DENY
  ipchains -P forward DENY
  ipchains -F

  #permit ssh
  for ipaddr in $SSH_PERMITTED;
  do
    ipchains -A input -p tcp -s $ipaddr -d 0/0 22 -i $LOCAL_INTERFACE -j
ACCEPT
  done

  # permit outgoing tcp
  ipchains -A output -p tcp -i $LOCAL_INTERFACE -j ACCEPT
  ipchains -A input -p tcp ! -y -i $LOCAL_INTERFACE -j ACCEPT
```

```
 # permit syslog
 for ipaddr in $SYSPERMITTED;
 do
   ipchains -A input -p udp -s $ipaddr -d $LOCAL_INTERFACE 514 -i
$LOCAL_INTERFACE -j ACCEPT
 done

 # if you would like to log all the other connection attempts,
 # uncomment these...
 #ipchains -A input -p tcp -i $LOCAL_INTERFACE -l -j DENY
 #ipchains -A input -p udp -i $LOCAL_INTERFACE -l -j DENY
 #ipchains -A input -p icmp -i $LOCAL_INTERFACE -l -j DENY
```

Appendix E

```
# pattern  to match on the first line
# action to perform on the second
# ==========================================


# Any bad login attempts
watchfor       /FAILED/
               echo bold

# Any attempts with sshd connections
watchfor       /SSHD/
               echo bold

# Any sniffing attempts
watchfor       /promiscuous/
               echo bold

# Any attempts login related activity
watchfor       /PAM_pwdb/
               echo bold
```

Appendix F

```
#!/bin/sh
 #---------------------------------
# Author: David A. Ranch
# Based on the TrinityOS file permissions corrections
 #---------------------------------
 MODE="o-rwx"
# Files in /bin
cd /bin
chmod $MODE linuxconf mount mt setserial umount
# Files in /sbin
cd /sbin
chmod $MODE badblocks ctrlaltdel chkconfig debugfs depmod dump*
chmod $MODE fdisk fsck* ftl* halt hdparm hwclock if* init insmod isapnp
chmod $MODE kerneld killall* lilo mgetty mingetty mk* mod* netreport
chmod $MODE pam* pcinitrd pnpdump portmap quotaon restore runlevel
chmod $MODE stinit swapon tune2fs uugetty # Files in /usr/bin cd /usr/bin
chmod $MODE control-panel comanche eject gnome* gpasswd kernelcfg
chmod 755 lp*
chmod 4755 lpr
#NOTE: I feel setting "lpr" to allow any group to execute it is
# a bad thing. #
# I would like to add UNIX users and even the Samba process to
# the "lp" group already defined in /etc/groups and then be able
# to put things back to 4750. BUT.. this really isn't possible.
# Linux doesn't support multiple groups per file and Linux
# doesn't support access lists (ACLs') yet. So.. you either have
# either leave these files SUID or run LPRng.
chmod $MODE minicom netcfg
 # Files in /usr/sbin
cd /usr/sbin
chmod $MODE at* crond dhc* edquota exportfs ftpshut group* grp*
chmod $MODE imapd in.* inetd ipop* klogd logrotate lp*
chmod 755 lsof
chmod $MODE makemap mouseconfig named* nmbd newusers ntp* ntsysv
chmod $MODE pppd pw* quota* rdev repquota rotatelogs rpc* samba
chmod $MODE setup showmount smb* squid syslogd taper tcpd* time*
chmod $MODE tmpwatch tunelp user* vi* xntp*
```

References

Securing Linux Step-By-Step *The SANS Institute*. Edited by Brotzman, Lee, E. and Ranch, David A. Version 1.0

Cole, E. Password Assessment and Management. *The SANS Institute GSEC Course Material*. Edited by Kolde, J. and Wendt, Karla

DePriest, Paul. Checklist for Securing Red Hat Linux 7.1 on an IBM Thinkpad Laptop**.** http://www.sans.org/y2k/practicals/Paul_DePriest_GCUX.zip. 8 NOV., 2001

Hines, Eric. “Complete Reference Guide to Creating a Remote Log Server**.”** http://www.linuxsecurity.com/feature_stories/remote_logserver-1.html. 22 AUG, 2000

Koconis, David “Step-By-Step Guide to Configuring an SSL enabled Web Server that Accesses a Backend Database using Red Hat 7.0.” http://www.sans.org/y2k/practicals/David_Koconis_GCUX.doc. 11 APR, 2001

Miller, Dave. “Time Synchronization Server” http://www.eecis.udel.edu/~ntp/ 26 NOV, 2001

Petersen, Bente. “Linux Red Hat 7.1 Security Assessment”**.** http://www.sans.org/y2k/practicals/Bente_Petersen_GCUX.zip. 31 AUG, 2001

Pitts, Donald. “Log Consolidation with Syslog.” http://www.sans.org/infosecFAQ/unix/syslog.htm 23 Dec, 2000

Pryor, Janice. “Installing and Securing a DNS/Mail Server Using Red Hat 7.1 Linux.” http://www.sans.org/y2k/practicals/Janice_Pryor_GCUX.zip. 12 AUG, 2001

Rudys, Algis. SSH Quick Start Guide http://linux.rice.edu/help/sshd ,23 Sept 2000

Sery, Paul. “Red Hat Linux 7.1 Installation Hardening Checklist.” http://www.sans.org/y2k/practicals/Paul_Sery_GCUX.rtf. 12 SEPT, 2001

Stevens, W. Richard. UNIX Network Programming . Englewood Cliffs, NJ: Prentice Hall 1990

Spitzner, Lance. “Watching Your Logs” http://www.linuxnewbie.org/nhf/intel/security/swatch.html No Date

Thomas, Benjamin D. “Creating Warning Banners.”

http://www.linuxsecurity.com/articles/network_security_article-631.html 11 May, 2000